Escola Politècnica Superior d'Enginyeria de Manresa

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Desenvolupament d'una aplicació mòbil per al Club d'Slot Llum Llamp

Treball final de Grau

Grau en enginyeria de sistemes TIC

Autor: Josep Nuño Farré
Tutora: Marta Isabel Tarrés Puertas
2017/2018

Dedicatòria

Aquest treball és el resultat de molts anys d'estudi i aprenentatge al llarg de la meva vida. Un esforç que he viscut de la mà dels meus pares, els quals sempre han fet el possible per ajudar-me en la meva formació acadèmica. També, al meu tiet Rafael Benito, el qual, va introduir-me dins d'aquest magnífic món de l'Slot (*Scalextric*), totalment desconegut per mi, ajudant-me sempre que sorgia un problema. Per últim, volia dedicar el treball a Llum Llamp Club Slot per tot el suport que he rebut al llarg d'aquests anys.

Agraïments

En primer lloc volia donar gràcies a la meva tutora del treball, Marta Isabel Tarrés Puertas, la qual, m'ha ajudat molt orientant-me tots els dubtes que m'han anat sorgit durant el projecte.

En segon lloc, a David Bruch que ha sigut el meu company de pràctiques des de segon any. Amb ell sempre he tingut molt bona comunicació, gràcies això, ens ha permès assolir els coneixements i avançar fins a quart.

Pròleg

L'Slot, és un esport competitiu que consisteix en fer córrer cotxes modelats (reduïts a una escala) i equipats amb motors elèctrics per uns circuits a la màxima velocitat possible. Els circuits estan formats per pistes les quals tenen una ranura per on circula el cotxe, aquestes estan alimentades elèctricament. Gràcies a això, s'ha batejat aquest món com a Slot. Tot i que aquest és el nom tècnic, molta gent l'associa a la marca *scalextric*, la qual, ha tingut molt pes en la indústria.

Aquest treball ha sorgit per facilitar moltes de les necessitats per un pilot i també per un club on es realitzan curses.

Llum Llamp, és un club d'Slot situat a la ciutat de Manresa, el qual, organitza curses durant tot l'any. Jo, formo part d'aquesta escuderia des de 2008 quan el meu tiet, Rafael Benito em va inscriure a la meva primera cursa.

Les curses fins ara, s'organitzaven enviant correus electrònics o amb trucades telefòniques, però, aquesta via és molt costosa i poc automatitzada. Analitzant la situació és on va néixer la idea d'aquest projecte.

A més a més, a partir d'aquest moment els pilots disposaran d'una aplicació que els permetrà gestionar configuracions dels comandaments, consultar els horaris, inscriure's a les curses...

Resum

Actualment resulta impensable un dia a dia sense l'ús del telèfon mòbil. Aquest resulta molt útil alhora de resoldre tasques o facilitar-nos la feina. Android es el sistema operatiu per a dispositius mòbils i tablets que mes ha crescut en els últims anys. Per aquest motiu, s'ha triat aquesta plataforma on implementar l'aplicació.

El projecte consisteix en el desenvolupament d'una aplicació totalment ajustada per al club d'Slot Llum Llamp. S'ha utilitzat l'entorn de treball Android Studio.

L'aplicació disposa de funcionalitats com: inscriure's a les curses, gestionar comandaments, consultar resultats, revisar el calendari... Per tal de millorar i facilitar la feina al club i als pilots. D'aquesta manera s'aconsegueix millorar la relació entre aquests dos col·lectius i promocionar les curses que hi ha durant tot l'any.

Paraules clau:

- Slot
- Aplicació Android
- Llum Llamp Club Slot
- *Scalextric*
- Mòbil
- Curses

Abstract

Nowadays it's unthinkable to imagine our life without the use of mobile phones. This object is really helpful because can solve and simplify our task. Android is the Operating System of mobiles and tablets which has grown up more in the last few years. For this reason, that's the platform where the application will be implemented.

The project consists in the development of an adjusted application for Llum Llamp Club Slot. The working environment was Android Studio.

The application got some functionalities like: sign up for a race, manage the slot commandments, check results and calendar... In order to improve and facilitate the work for the club and drivers. In this way, it's possible to improve the relationship between these two groups and promote the races that exist throughout the year.

Keywords:

- Slot
- Android Application
- Llum Llamp Club Slot
- *Scalextric*
- Mobile
- Race

Índex de contingut

Índex d'il·lustracions

Índex de taules

1. Introducció

En l'actual era moderna, tothom fa servir el telèfon mòbil per a tot tipus d'ús, navegar per internet, realitzar pagaments, veure vídeos, escoltar música… Per a molta gent, és un objecte imprescindible dins de la seva vida quotidiana. El fet de poder crear aplicacions mòbils de manera lliure en entorns *OpenSource* és una gran eina de treball per a donar solucions a problemes o necessitats molt concretes.

En el món de l'Slot, on hi ha curses pràcticament cada cap de setmana i on s'ha de gestionar les assistències dels participants de manera eficaç, fer servir una eina comuna, com podria ser una aplicació mòbil, és una manera ràpida i simple de solucionar un problema de comunicació que pot portar molts mals de cap. La falta d'automatisme d'aquest procés, pot suposar un desordre i una gran inversió de temps per organitzar manualment les curses.

A més a més, fer una aplicació mòbil també ofereix solucions a necessitats dels corredors. A qui no li agradaria seguir en directe els resultats de la cursa? O poder gestionar les configuracions dels comandaments? O fer fitxes tècniques dels seus vehicles? O fins i tot, una cosa tan simple com mantenir-se informat de les notícies…

Al llarg d'aquest treball es tractarà el disseny de l'aplicació per Android, esmenant les eines de treball utilitzades, la implementació de la solució i desenvolupament d'aquesta. Cal destacar l'ús de vocabulari tècnic informàtic i de l'Slot, el qual, està englobat dins de dos glossaris que es poden trobar a l'annex del treball.

1.1 Motivacions

La programació amb Java, un llenguatge molt semblant a C però amb expressions molt abstractes properes a Python sempre m'ha cridat l'atenció, tot i que, al llarg del grau encara no l'havíem utilitzat, m'ha servit per aprendre moltíssim d'aquest llenguatge. El disseny d'aplicacions per Android és un context molt bo per practicar Java i resoldre el principal problema que està proposat. També, fins aquest moment no havia programat mai amb PHP i realitzant el *webservice* m'he pogut iniciar àmpliament.

A més a més, el fet de practicar aquest *hobby* i estar molt implicat en aquest món, m'ha aportat moltes ganes de dur a terme alguna cosa relacionada amb això. També, ajudar al club on sóc soci, per tal de posar les coses més alhora d'organitzar carreres i gestionar els seus participants.

1.2 Objectius

El principal objectiu del treball és desenvolupar una **aplicació** accessible per a qualsevol usuari **d'Android, senzilla, eficient i ajustada**.

Aquesta aplicació ha de permetre resoldre totes les necessitats tant del club com del pilot que participa en les seves curses. **L'aplicació ha de tractar els següents punts:**

1. **El primer objectiu** és permetre **la inscripció a les curses** al llarg de l'any de manera ràpida i senzilla mitjançant l'aplicació.
2. **El segon objectiu** és facilitar a l'usuari la capacitat d'emmagatzemar i **gestionar** la informació relacionada amb les **configuracions dels comandaments d'Slot.**
3. **El tercer objectiu** és permetre a l'usuari **generar fitxes tècniques** amb les referències del material utilitzat en cada component (llantes, politges, eixos...) per tal de tenir anotat quin material porta el seu cotxe.
4. **El quart objectiu** és permetre **seguir en directe (*streaming*) els resultats** dels pilots que participen en la cursa.
5. **El cinquè objectiu és millorar la comunicació** que hi ha entre l'escuderia i el pilot. Aconseguir que l'usuari es trobi integrat dins del club i conegui totes les novetats del dia a dia.

També seria molt interessant poder penjar l'aplicació a la **Play Store** i desenvolupar el sistema no únicament a escala local, sinó que sigui accessible per a tothom.

1.3 Justificació, perquè Android?

Android és un sistema operatiu basat en el ***kernel* de Linux** i dissenyat principalment per a dispositius mòbils amb pantalla tàctil, tals com *smartphones* o tablets. Inicialment va ser un projecte d'Android INC, però, a partir de 2005, Google va adquirir l'empresa.

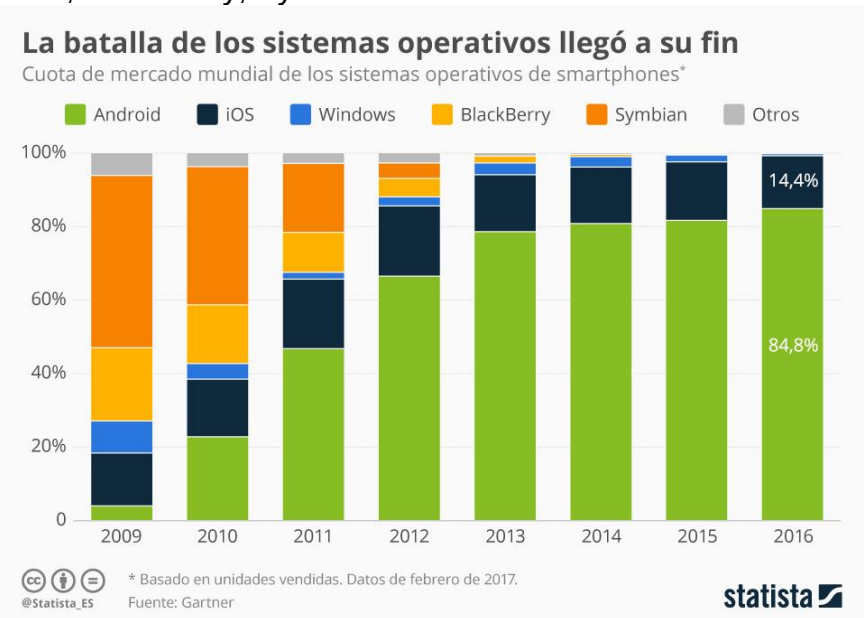
L'aspecte més fonamental per arribar al punt actual d'expansió d'Android va ser desenvolupar un sistema operatiu multiplataforma, ja que fins fa poc, un sistema operatiu s'associava únicament a un dispositiu hardware. Amb aquesta idea de desenvolupament, s'ha convertit en el **S.O més utilitzat segons el portal web d'estadístiques *statista***. La data de consulta és de febrer de 2017.

El sistema disposa de diverses versions que segueixen l'abecedari, arribant a la lletra P.

LLETRA	NOM	VERSIÓ	% USUARIS
A	*Apple Pie*	1.0	1%
B	*Banana Bread*	1.1	
C	*Cupcake*	1.5	
D	*Donut*	1.6	
E	*Éclair*	2.0/2.1	
F	*Froyo*	2.2	
G	*Gingerbread*	2.3	
H	*Honeycomb*	3.0...3.2	
I	*Ice Cream Sandwich*	4.0	
J	*Jelly Bean*	4.1...4.3	6,2%
K	*KitKat*	4.4	13,8%
L	*Lollipop*	5.0/5.1	**27,2%**
M	*Marshmallow*	6.0/6.0.1	**30,9%**
N	*Nougat*	7.0....7.1.2	**20,6%**
O	*Oreo*	8.0/8.1	0,3%
P	*P*	9.0	Desconegut

Taula 1: Versions d'Android disponible. FONT: Externa[24]

Aquest sistema domina a nivell mundial, de manera imponent amb rivals com iOS, Windows, BlackBerry, Symbian...



Il·lustració 1 : Batalla de sistemes operatius mòbils. FONT: Externa[23]

En aquesta gràfica es pot veure clarament que si l'aplicació s'ha d'adreçar al nombre més gran de persones, el S.O a utilitzar es Android. També, s'observa clarament que Android ha absorbit la població que utilitzava Symbian i BlackBerry. Mentrestant el públic d'iOS es manté fidel amb el 13-16% d'usuaris al llarg dels anys.

2. Estat de l'art

En el segon capítol, es realitza una recerca de quines aplicacions es poden trobar al mercat i l'anàlisi de les seves característiques. També, es valora la qualitat d'aquestes i s'extrauran idees pel projecte. Per acabar, es proposa un primer enfocament de com hauria de ser l'aplicació.

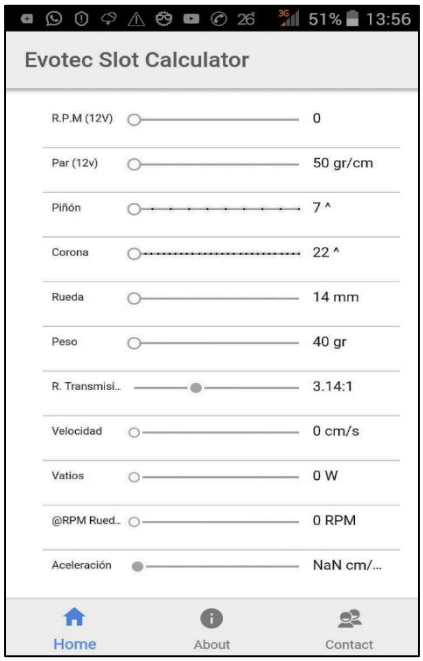
2.1 Aplicacions existents

En el món de l'Slot no es troben gaires aplicacions semblants a la desenvolupada. Tot i així, el servei està més enfocat a nivell web. La **majoria de clubs** disposen de **pàgines web** on es pot realitzar **preinscripcions a les curses**. També és on els competidors poden **consultar els resultats i veure les classificacions del campionat**.

Actualment es troben publicades a la Play Store de manera **gratuïta** dues aplicacions, **Evotec Slot Calculator i Slot Car Diary**.

EVOTEC Slot Calculator: Aquesta aplicació està desenvolupada pel distribuïdor de material de competició Evotec. Aquesta empresa està molt valorada dins de l'àmbit de competició perquè permet al seu client comprar material de diferents marques a través d'internet i rebre'l còmodament a casa.

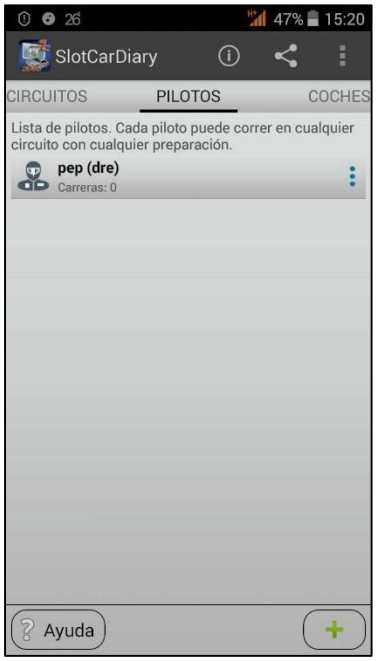
L'aplicació que han desenvolupat està **centrada en l'àmbit de les curses de Velocitat**. Consisteix en calcular diferents paràmetres tècnics del vehicle en funció de les R.P.M del motor, relació de transmissió, pes... per tal d'obtenir una velocitat màxima, acceleració...



Il·lustració 2 : App Evotec Slot Calculator. FONT: Pròpia

SLOT CAR DIARY: Aplicació desenvolupada per David Benito León. **Disposa de funcionalitats com gestionar la informació dels cotxes, circuits, pilots, preparacions, recanvis i carreres**. L'aplicació actua com un diari on es van anotant els canvis del dia a dia.
Aquestes imatges estan extretes de l'aplicació:

Il·lustració 3 : Slot Car Diary I
FONT: Pròpia



Il·lustració 5 : Slot Car Diary II
FONT: Pròpia

Il·lustració 4 : Slot Car Diary III
FONT: Pròpia

La primera imatge correspon a la pantalla inicial. On es pot puntuar o accedir al menú.

En el cas de la segona, es pot veure el menú per on es pot desplaçar l'usuari i gestionar els circuits, pilots, cotxes... i afegir-ne de nous.

La tercera imatge correspon al registre d'un cotxe, on s'observa, com cada usuari afegeix els seus cotxes omplint els paràmetres de: nom, escala, marca, model, any, data de compra i inclús una fotografia.

La primera aplicació de la casa **Evotec**, només disposa d'una **única finestra** on pots realitzar els càlculs matemàtics. Per tant, és molt reduïda i simple. Tot i això, és extremadament útil, ja que d'una manera eficaç, obtens una resposta exacte i científica per a cada petit canvi de relació de transmissió, voltatge... sense haver de fer un càlcul a mà.

La segona, **Slot Car Diary**, és una aplicació molt ambiciosa. Actualment encara **està en desenvolupament (versió alfa)** perquè no té disponibles algunes funcionalitats, com podria ser, afegir noves categories de manera personalitzada als recanvis, quan l'usuari ho intenta fer, apareix un missatge en forma de *Toast.message* que indica *"próximamente."*

Il·lustració 6 : Slot Car Diary IV. FONT: Pròpia

Fins a l'actualitat, disposa de moltes opcions útils per tal de fer la vida més fàcil al pilot, però, no disposa d'una funcionalitat que permeti **gestionar** les **configuracions dels comandaments**.

2.2 Proposta

La proposta consisteix a realitzar una aplicació que permeti millorar **la relació club-pilot**. Dins de l'aplicació es podran incloure funcionalitats semblants a les que es poden trobar a l'*app* Slot Car Diary per a fer-la més atractiva a un públic més gran, és a dir, un usuari que no és habitual del club, podria acabar utilitzant l'aplicació gràcies a les múltiples funcionalitats útils incloses a l'aplicació.

A més a més, l'aplicació ha de permetre facilitar la gestió de curses del club i millorar la interacció amb els usuaris. D'aquesta manera, socis, no-socis i corredors estaran més informats de les novetats.

3. Etapa d'anàlisis

En aquest capítol del projecte es presenta una primera fase d'anàlisis, on es defineixen els requisits i es mostra de manera global, l'arquitectura que es vol proposar com a solució del problema. És en aquest apartat on s'ha d'assentar les bases i fonaments del projecte

S'especificarà el conjunt de requeriments que ha de tenir com a mínim de manera funcional i altres de complementaris que augmentin la qualitat del producte. També, s'establirà la metodologia de treball adient per ajustar-se al projecte.

L'arquitectura de l'aplicació serà en format client-servidor. Estarà format per un servidor on estarà allotjada una base de dades amb una API d'accés i per últim l'aplicació mòbil que utilitzarà el client.

També es comentarà com haurà d'estar estructurada la base de dades i una breu recerca sobre la seguretat de l'aplicació.

3.1 Catàleg de requeriments

Part essencial que delimita fins a quin punt arribaran les funcionalitats i la qualitat de l'aplicació. Observant l'estat de l'art s'ha establert una sèrie de requisits indispensables pel projecte.

3.1.1 Requeriments funcionals

Els requeriments funcionals descriuen totes les interaccions que haurà de tenir els usuaris amb el software dissenyat.

Gestió del compte d'usuari:

A continuació s'exposaran els requeriments associats a múltiples accions que es podran realitzar dins del compte d'usuari, per exemple, un procés de registre, d'identificació, edició del perfil, inscripció...

Registre:

- L'aplicació ha de permetre registrar-se mitjançant un formulari.
- El sistema encarregat de tractar les dades ha de ser capaç de validar-les i informar a l'usuari si alguna de les dades introduïdes no són correctes.
- En cas de que tots els paràmetres siguin correctes, automàticament el sistema haurà d'emmagatzemar les dades en una base de dades.
- Per verificar que el correu utilitzat en el registre és realment de l'usuari, s'haurà d'enviar un missatge de verificació amb un enllaç d'activació.
- Un cop completat el registre, el sistema haurà de donar la benvinguda.

Log in/ Identificació:

- L'usuari haurà d'iniciar sessió amb les seves credencials (usuari i contrasenya).
- En cas que l'usuari hagi iniciat sessió anteriorment i l'hagi deixat oberta, no necessitarà introduir les seves dades de nou, el sistema ho gestionarà automàticament.
- Si l'usuari no introdueix les seves credencials correctament, automàticament mostrarà un missatge d'error.

Log Out/Tancament de sessió:

- L'usuari ha de poder tancar la sessió.
- Un cop es tanqui la sessió, l'usuari serà redirigit a la pantalla de Login per si vol iniciar sessió amb un altre compte.

Perfil i dades de l'usuari:

- L'aplicació ha de tenir una pantalla o varies capaces d'administrar les dades de l'usuari.
- El sistema ha de permetre actualitzar paràmetres com per exemple, nom, cognom, correu, *nick*...
- El sistema ha de comprovar que tots els paràmetres introduïts són correctes. Si ho són, s'actualitzarà la informació a la base de dades.
- En cas d'error, s'informarà de quin camp és l'erroni i el motiu.
- L'aplicació ha de proporcionar algun **sistema de recuperació del compte** en cas que l'usuari oblidi la contrasenya, el *nickname* o el compte hagi sigut robat.
- En cas que es vulgui canviar l'adreça de correu, s'haurà d'enviar un correu de confirmació a l'e-mail antic. Un cop acceptat, s'haurà d'enviar un missatge al nou correu per tal de verificar que és de l'usuari.
- Si al canviar la direcció de correu, aquesta ja està en ús per un altre usuari, se l'informarà.
- L'usuari es pot donar **de baixa** posant-se en contacte amb el correu especificat pel club, **enviant un e-mail utilitzant el correu registrat a l'aplicació**.

Inscripcions:

- Haurà d'existir una o un conjunt de pantalles capaces de facilitar les inscripcions a les curses.
- El sistema ha de permetre consultar els inscrits a una cursa mitjançant un formulari.
- El sistema ha de permetre afegir una inscripció nova utilitzant un formulari.
- Si l'usuari introdueix algun paràmetre malament, se l'haurà d'informar. En cas contrari, es processarà una nova inscripció i s'emmagatzemarà a la base de dades.

Entrenament:

- Una o un conjunt de pantalles han de permetre gestionar les configuracions dels comandaments.
- L'usuari podrà visualitzar la llista de configuracions i editar-les de manera individual. A més a més podrà emmagatzemar-les i eliminar-les.

- Cada configuració del mando tindrà com a paràmetres: el nom de la configuració, voltatge, freno, acceleració i *anti-spin*.
- Si el nom de la configuració ja està en ús, ho informarà a l'usuari perquè la modifiqui.

L'aplicació:

L'aplicació estarà formada per uns elements bàsics com per exemple, una *splash screen* a l'iniciar l'aplicació, un menú principal que faci de guia a l'usuari i un menú desplegable que faciliti el moviment entre pantalles.

***Splash screen*:**

- Cada cop s'obri l'aplicació apareixerà durant un petit període de temps el logotip de LlumLlamp Club Slot. Posteriorment es donarà pas al menú principal.

Menú principal:

- La pantalla principal de l'aplicació constarà de 6 botons:
    a. **Calendari:** Permet consultar les curses.
    b. **Taller:** Formulari que permet generar fitxes tècniques del vehicle.
    c. **Inscripció:** Pantalla que permet afegir una nova inscripció o consultar inscrits.
    d. **Entrenament**: Lloc on es podran veure, afegir, editar... les configuracions del comandament.
    e. **Resultats:** Permetrà accedir de manera ràpida als resultats publicats a la web.
    f. **El meu Perfil:** Accés de manera ràpida a la pantalla on es troben totes les dades de l'usuari.

Menú desplegable lateral:

- El menú lateral haurà de tenir una capçalera on es veurà el nom de l'usuari i el compte de correu actual.
- En cas que l'usuari no estigui *loggejat*, apareixerà l'estat d'invitat.
- Ha de permetre accedir a totes pantalles que apareixen al menú principal.
- També, ha de permetre l'accés a:
    a. *Login*: Pantalla on es pot iniciar sessió.
    b. Registre Usuari: Formulari on registrar un usuari nou en l'aplicació.
    c. Menú: Pantalla principal de l'aplicació on es troba l'accés a les principals funcionalitats de l'aplicació.
    d. Calendari: Lloc on consultar les curses del club.
    e. Ubicació: Obrirà automàticament Google Maps amb la localització de Llum Llamp Club Slot.
    f. Contacte/ Horaris: Mostrarà la informació de xarxes socials que utilitza Llum Llamp i els telèfons, correus... A més a més, indica l'horari d'obertura del club.
    g. Cursa en directe: Obrirà el navegador amb el canal de Twitch on es retransmetrà la cursa en directe.

3.1.2 Requeriments no funcionals

Els requeriments no funcionals, o atributs de qualitats, són aquells requisits que es poden utilitzar per jutjar l'operació d'un sistema en comptes del comportament específic (requisits funcionals). Alguns exemples d'aquests són: rendiment, disponibilitat, accessibilitat, seguretat...

Documentació:

- Manual d'ús de l'aplicació.
- El codi escrit ha d'estar ben estructurat i comentat, d'aquesta manera un programador nou dins el projecte, podrà afegir contingut fàcilment.

Seguretat:

- Per poder utilitzar el 100% de l'aplicació has de *loggejar-te*.
- L'usuari podrà continuar amb la sessió oberta si no es *desloggeja*, per tant, no haurà d'autentificar-se cada vegada.
- Totes les contrasenyes seran emmagatzemades xifrades a la base de dades.
- Cada cop que l'usuari vulgui **restaurar el compte, activar un correu o canviar-lo, rebrà un codi únic per aquella petició**. Serà d'un sol ús.
- El servei web tindrà protegit els directoris amb un usuari i contrasenya.
- L'administrador del sistema podrà eliminar, editar o afegir usuaris, inscripcions, rallys... si és necessari.

Portabilitat:

- L'aplicació ha de funcionar en el màxim nombre de dispositius Android: tablets o mòbils.
- Totes les pantalles s'ajustaran a dispositius mòbils amb diferent resolució de pantalla, mida i versió.
- Per poder fer ús de l'aplicació al complert, haurà de tenir connexió a internet i estar registrat.

Ús e interfícies:

- S'ha de controlar totes les accions on l'usuari pot introduir dades per tal d'evitar errors de seguretat.
- L'aplicació ha d'estar formada per unes interfícies senzilles, atractives e intuïtives, d'aquesta manera, no suposarà un impediment o un esforç utilitzar l'aplicació.
- S'ha de controlar qualsevol tipus de *bug* o error que es pugui generar de manera malintencionada en el servei web o en l'aplicació.

Rendiment:

- L'usuari ha de desplaçar-se entre pantalles sense patir bloquejos al carregar la IU. A més a més, la interacció amb els elements que formen la IU haurà de ser fluida sense patir cap tipus de *lag*.

3.2 Metodologia

Per tal de poder resoldre els principals objectius de manera eficaç i ordenada, s'haurà de seguir una sèrie de passos. En cadascun d'ells, es provaran els diferents prototips fins a arribar al producte final:

1r pas: Formació/aprenentatge de coneixements. **Investigar** com es realitza el **desenvolupament d'aplicacions per Android**. També, aprendre les bases del llenguatge de **programació Java**. D'aquesta manera s'assentaran les bases/pilars del projecte.

2n pas: Aconseguir **un primer prototip de l'aplicació**. Aquesta ha d'estar formada per diferents fragments i activitats (pantalles) que donin solució als principals problemes proposats.

3r pas: **Incorporar una base de dades local**. L'objectiu és obtenir un projecte bastant semblant al definitiu. Aquest haurà de gestionar una base de dades idèntica a la que farem servir de manera remota en el servidor.

4t pas: Desenvolupar un *webservice* local capaç de gestionar la nostra base de dades. Aprendre coneixements de **PHP i JSON**, com funcionen/utilitzen aquestes eines. A més a més, aprendre les comandes disponibles amb PHP per gestionar una base de dades amb **MySQL**.

5è pas: Assentar una comunicació estable i totalment *debuggada*. Realitzar comunicacions via **peticions HTTP POST/GET** utilitzant la **llibreria Volley**, la qual, ens permetrà comunicar-nos amb el nostre servei web.

6è pas: Desenvolupar el ***webservice* en un domini públic** per tal d'aconseguir sortir de la xarxa local.

Cadascun d'aquests apartats establiran punts d'inflexió en el treball. També s'utilitzaran com a punts de restauració si sorgeix algun problema. Com a últim pas, aconseguir distribuir l'aplicació perquè qualsevol persona hi tingui accés publicant-la a la Play Store.

3.3 Bases de dades

Les bases de dades són unes eines imprescindibles per gestionar la informació. Per dur a terme aquest projecte, seran necessàries dues bases de dades:

La **base de dades local**, es trobarà en el dispositiu mòbil de l'usuari, s'utilitzarà per gestionar les configuracions del comandament, d'aquesta manera, s'evita moltíssim tràfic d'informació quan l'usuari realitza un canvi en les configuracions. Conseqüentment, suposa un **avantatge molt important** perquè no serà necessari tenir disponibilitat d'una xarxa (ja pot ser utilitzant el servei de dades o *Wifi*) per gestionar les configuracions.

Per exemple, si un club d'Slot es troba en un soterrani d'un edifici, l'usuari, pot perdre la cobertura, lo qual suposaria que no podria gestionar les configuracions, ni tan sols consultar-les durant els entrenaments o la cursa.

Per solucionar aquest problema de connexió, s'ha desenvolupat el **mode desconnectat.** D'aquesta manera, si l'usuari no pot *loggejar* per problemes de cobertura, podrà accedir a gestionar els comandaments **sempre que, no hagi tancat sessió anteriorment.**

Mentrestant, **la base de dades remota**, emmagatzemarà la informació sobre els *rallys* disponibles, inscripcions, usuaris, cotxes registrats, peticions de seguretat... Aquesta serà gestionada per un administrador del club.

3.4 Seguretat

L'estudi de la seguretat és imprescindible per identificar perills i eliminar-los. Per aquest motiu, a continuació, s'analitzarà els perills que poden sorgir en la comunicació, accés a la base de dades remota i la seguretat de la informació a la base de dades.

3.4.1 Comunicació

L'aplicació es comunica mitjançant peticions HTTP amb el servidor, per rebre i obtenir dades. La informació pot veure's en perill vers atacs *Man in the Middle*. Si l'atacant és capaç d'interceptar la informació de les capçaleres on s'especifica l'usuari i contrasenya d'autentificació al *webservice*, podria tenir accés a executar algun *script* PHP, però, sense conèixer l'estructura interna no aconseguiria fer un atac efectiu.

3.4.2 Accés a la base de dades remota

L'accés a la base de dades es fa mitjançant petits scripts amb PHP, els quals gestionen les peticions i reben els paràmetres necessaris per realitzar la consulta, inserció o actualització de la informació. Aquests **paràmetres han d'estar tractats prèviament** per tal d'evitar **injeccions SQL**.

3.4.3 Seguretat de la informació a la base de dades

A continuació es tractaran els principals algoritmes que s'utilitzen en l'actualitat per emmagatzemar informació encriptada a les bases de dades. S'exposarà una descripció, un exemple i els punts forts de cadascun d'ells. Després s'explicarà quina és la funció de l'algoritme d'encriptació MD5 i BLOWFISH dins del treball.

MD5

MD5 és un algoritme de reducció criptogràfic de 128 bits utilitzat actualment arreu del món. Va ser desenvolupat per Ronald Rivest l'any 1991. Van fer falta 5 anys, quan Hans Dobbertin va aconseguir trencar l'algoritme parcialment publicant que es produïa una col·lisió de *hash* (valors d'entrada diferents produeixen una mateixa sortida) generant moltíssima controvèrsia a sobre del seu futur. Tot i això, aquest algoritme és extremadament ràpid alhora de transformar informació.

SHA

És una funció de *hash* creada per l'Institut Nacional de Normes i Tecnologia (NIST) l'any 1993. La versió que s'havia plantejat utilitzar, SHA-1, va ser trencada parcialment l'any 2004, on es va trobar una debilitat matemàtica que permetia trobar col·lisions de *hash* de manera eficient. Cal destacar que és un mètode lleugerament més seguir que md5, però, alhora més lent al transformar la informació.

Tot i que en l'actualitat aquests algoritmes (md5 i sha1) encara s'utilitzen perquè no és del tot trivial desencriptar-los mitjançant força bruta, en els pròxims anys veurem com les noves màquines amb més capacitat de càlcul ho aconseguiran.

Per aquest motiu, s'utilitza la llibreria de PHP per emmagatzemar contrasenyes amb l'algoritme BCrypt (implementació de PHP del algoritme Blowfish).

BLOWFISH

En criptografia, **Blowfish** és un codificador de blocs simètric, dissenyat l'any 1993. Fins a l'actualitat no s'han trobat tècniques de criptoanàlisis efectives contra aquest algoritme. Aquest utilitza blocs de 64 bits i claus que van des de 32 bits a 448 bits. És un codificador de 16 rondes Feistel i utilitza claus que depenen de Caixes-S (component bàsic dels algoritmes de xifrat amb clau simètrica. Transformen un nombre de bits M en un conjunt N, MxN). Té una estructura semblant a CAST-128, però, aquest utilitza Caixes-S fixes.

Contrasenyes:

Les contrasenyes no poden ser visibles per cap persona que gestioni la base de dades, per aquest motiu, és imprescindible que estiguin encriptades. Com s'ha comentat anteriorment, aquestes s'emmagatzemen encriptades **mitjançant l'algoritme BCrypt de PHP**.

ALGORITME	PARAULA	HASH
MD5	Llumllamp	F1A0122720671B75FAB390841E692E59
SHA-1	Llumllamp	923dbb374a67f6939732e82073eb8f1d7d671159
BCRYPT	Llumllamp	$2y$12$DtT7g9/B8KZJAIkgpCwX7.tEisN3ua8YaVLzDFjsH8hYx1whDEi9O

Taula 2 : Exemple encriptació contrasenyes amb els diferents algoritmes. FONT: Pròpia

Codi d'activació, confirmació de canvi de correu i restauració:

Quan l'usuari es registra a l'aplicació genera un codi d'activació, tanmateix quan es realitza una petició de seguretat de canvi de correu i restauració.

Aquests **codis s'incrustaran al enllaç** que s'envia al correu del usuari per a que confirmi la petició. Amb el codi generat, només podrà realitzar 1 cop la sol·licitud corresponent. Automàticament caducarà. El codi generat no pot ser aleatori perquè ha de ser totalment únic, per això, s'obtindrà mitjançant l'algoritme **md5** (correu_usuari+horactual).

4. Disseny

En el següent capítol es descriu el procés de disseny de l'aplicació Llum Llamp Club Slot. En el disseny s'ha englobat tots els requisits especificats en el 3.1 Catàleg de requeriments.

El capítol es divideix en diversos apartats on es tractarà les eines de treball, les llibreries utilitzades, l'arquitectura global del disseny, l'arquitectura del software de l'aplicació, les bases de dades, la recepció de peticions al servidor i la seguretat.

4.1 Eines de treball

A continuació es tractaran les principals eines de treball utilitzades al llarg del projecte. El llistat és el següent:

- Android Studio
- XAMPP
- 000webhost
- SQLite
- MySQL
- SQLyog
- Twitch
- OBS STUDIO

S'exposarà en cadascuna d'elles **el seu logotip, la funcionalitat, la descripció tècnica i a més a més l'enllaç de descàrrega**.

ANDROID STUDIO

Il·lustració 7 : Logo Android Studio. FONT: Externa[25]

PLATAFORMA: Ubuntu 16.04 LTS VERSIÓ: 3.1.2. Build #AI-173.4720617, abril 13, de 2018. JRE: 1.8.0_152-release-1024-bo1 amd64. JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o. És pot trobar en aquest enllaç.

És l'entorn de desenvolupament (IDE) oficial per crear aplicacions per Android, basat en IntelliJ IDEA (IntelliJ és capaç de donar suggeriments rellevants alhora d'escriure el teu codi en cada context). A més a més del potent editor de textos i eines de IntellIJ, Android Studio ofereix moltes funcions capaces d'augmentar la capacitat d'aconseguir codi productiu.

Disposa de funcions com les següents:

- Sistema de compilació basat amb el *Gradle* flexible (sistema d'automatització de construcció de codi obert que construeix sobre conceptes de *Apache Ant i Apache maven*).
- Emulador ràpid i eficient.
- Compatibilitat amb C++ i NDK.
- Renderització en temps real.
- Refactorització específica d'Android.
- Consola de desenvolupadors.
- Suport incorporat per Google Cloud Platform.
- *Instant run* d'emuladors per tal d'executar la teva aplicació...

L'estructura del projecte en Android es pot dividir en 3 parts:

- **manifest**: L'arxiu AndroidManifest.xml conté la informació essencial de l'aplicació, aquesta informació l'utilitzarà el sistema Android per tal de poder executar-la. A més a més, és capaç de descriure el paquet de Java que utilitza, les diferents pantalles que la integren, declara els permisos necessaris per utilitzar tots els components de l'aplicació, declara la versió mínima i màxima d'Android necessàries, les biblioteques vinculades...
- **java:** Conté el codi font de Java que serveix per descriure les interaccions dels *widgets*, botons, *containers*... També conté la informació d'altres classes no associades a la interfície gràfica.
- **res:** Conté tots recursos, com per exemple: dissenys XML, imatges, els *Strings* de la interfície gràfica (IU)... Per tant, és on es trobarà el contingut gràfic de l'aplicació.

XAMPP(X: qualsevol S.O, A:apache, M:MySQL, P: PHP, P:Perl)

Il·lustració 8 : Logo XAMPP. Font: Externa[26]

PLATAFORMA: WINDOWS 7. VERSIÓ: 7.2.4 (12 abril de 2018). Es pot trobar el programari en el següent enllaç.

XAMPP és un paquet de software lliure, format principalment per un gestor de base de dades **MySQL**, un servidor web **Apache** i intèrprets de *scripts* **PHP** i **Perl**. Es distribueix amb llicència GNU i actua com a servidor web lliure. Es troba disponible per Microsoft Windows, GNU/LINUX, Solaris i Mac OS X. Ha estat desenvolupat per *Apache Friends*. Aquest servei disposa de:

- suport de les **versions de PHP**: 7.2.4, 7.1.16, 7.0.29 i 5.6.35.
- **MySQL 5**
- **OpenSSL 1.0.2g**
- **phpMyAdmin 4.8.0**
- **Apache 2.4.33**

La seva funcionalitat dins del projecte, és veure el comportament de l'aplicació **dins d'un servei local**. D'aquesta manera es pot testejar la comunicació amb la llibreria Volley i els documents PHP corresponents. Un cop l'aplicació treballi correctament de manera local, s'haurà de hostejar en un domini públic.

000WEBHOST

Il·lustració 9 : Logo 000Webhost. Font: Externa[27]

000webhost és un proveïdor d'allotjament (*hosting*) de serveis web totalment **gratuït.** El servei web ofereix aquestes prestacions:

Preu mensual	Gratis
Espai de disc disponible	1000Mb
Ample de banda	10Gb
Serveis web allotjats	Màxim 2
Panell de control	Gratis
Serveis publicitaris propis	Gratis
Domini públic	Gratis
Allotjament de WordPress	Gratis
Temps de manteniment (*sleep time*)	1hora/per dia

Taula 3 : Serveis 000webhost. FONT: Externa[41]

A més a més, permet utilitzar bases de dades **MySQL, suport FTP, gestor d'arxius a nivell web, recuperacions d'arxius (backups), curl, Cron Jobs i les últimes versions de PHP (7.2).** Cal destacar que **té protecció hotlink**, és a dir, evita que altres serveis web siguin capaços de vincular qualsevol arxiu al teu *webservice*. També, **permet protegir els directoris del servei web** amb una autentificació.

La gestió de la base de dades es pot manegar de manera extremadament còmode mitjançant la interfície **de phpMyAdmin versió 4.7.7. La versió de PHP actual és la 7.0.26. El servidor web** és un servidor **Apache/2.4.6 (CentOS) que utilitza OpenSSL/1.0.2k-fips** com a protocol de seguretat en la comunicació per la red**.**

OpenSSL és una llibreria escrita en C utilitzada per la majoria de serveis webs la qual conté una implementació de codi obert del protocol **TLS i SSL.**
El servidor de la base de dades es torba via **UNIX socket,** és un servidor tipus **MariaDB versió 10.1.31 amb suport UTF8 Unicode.**

SQLITE3

Il·lustració 10 : Logo SQLite. FONT: Externa[28]

PLATAFORMA: Ubuntu 16.04 LTS VERSIÓ: 3.11.0 És pot trobar en aquest enllaç.

SQLite és un sistema de gestió de bases de dades relacionals compatible amb transaccions de **tipus ACID**, continguda en una relativa petita biblioteca escrita en C. És un projecte de domini públic creat per D.Richard Hipp. Android Studio pot suportar la gestió de bases de dades amb SQLite, el qual, permet tenir una base de dades local en les aplicacions.

SQLite serà l'entorn on es desenvoluparà el primer prototip de la base de dades. Permetrà fer les proves de consistència i disseny de taules.

MYSQL

Il·lustració 11 : Logo MySQL. Font: Externa[29]

MySQL és un sistema gestor de base de dades relacional servides a nivell web desenvolupat en llicencia dual: Llicència pública i llicència Comercial per Oracle Corporation. És considerada com la base de dades de "codi obert" més popular del món. Actualment el motor InnoDB (utilitzat en el disseny), és el més utilitzat per gestionar MySQL.

El motor **InnoDB** és un motor d'emmagatzemament de dades de codi obert per a MySQL. S'inclou com a format de taules estàndard per totes les versions de 4.0 en amunt. La seva característica principal és que suporta **transaccions de tipus ACID** (atomicitat, consistència, aïllament i durabilitat) i bloqueig de registres e integritat referencial. Ofereix una fiabilitat i consistència bastant superior a MyISAM, que era la tecnologia que s'utilitzava en les taules de MySQL incapaç de suportar transaccions. La versió que utilitza XAMPP de **MySQL server és la 5.5. 000webhost suporta les versions 5.+**

SQLYOG

Il·lustració 12 : Logo SQLyog. FONT: Externa[30]

PLATAFORMA: WINDOWS 7 VERSIÓ: 13.0.0 (64 bits). Es pot descarregar de manera gratuïta en aquest enllaç.

SQLyog és un GUI desenvolupat per Webyog bastant popular per gestionar bases de dades en MySQL sobre el sistema operatiu de Windows. Actualment ha sigut descarregat més de 2,5 milions de vegades arreu del món.
S'utilitza principalment per a gestionar la base de dades de manera local amb Xampp Server.

TWITCH i OBS STUDIO

Il·lustració 14 : Logo Twitch. FONT: Externa[32]

Il·lustració 13 : Logo OBS STUDIO. FONT: Externa[31]

Twitch és una plataforma que ofereix servei de *streaming* de vídeo en viu. Actualment està sota propietat d'Amazon. En aquesta plataforma es retransmeten jocs, eSports i esdeveniments relacionats amb videojocs. El contingut pot ser sota demanda o en directe.

Per tal de tenir un espai dins de la plataforma, s'ha creat un canal amb nom: **llum_llamp_club_Slot** .

L'eina encarregada de poder retransmetre el contingut capturat des de l'ordinador és **OBStudio** (*Open Broadcast Software*). És una aplicació lliure i de codi obert per a la captura i transmissió de vídeo. Suporta el sistema operatiu de Linux, MacOSx i Windows. Aquesta està escrita en C i C++ i permet capturar fonts de vídeo en temps real, composició en escena, codificació, captura i retransmissió. Utilitza el protocol **RTMP** i la plataforma destí pot ser Youtube, Twitch, DailyMotion entre altres.

PLATAFORMA: WINDOWS 7. VERSIÓ:21.1.0 (13 maig 2018) . L'enllaç de descàrrega es troba aquí.

4.2 Llibreries externes utilitzades

Les llibreries externes han sigut imprescindibles per dur a terme varies de les funcionalitats. Algunes d'aquestes no es trobaven integrades dins del *framework* d'Android o PHP i s'han hagut d'importar. El llistat és el següent:

- PHPMailer
- iTEXTPDF
- Volley
- Password hash
- MYSQLi

S'exposarà en cadascuna d'elles **el seu logotip, una breu descripció tècnica i a més a més l'enllaç de descàrrega**.

PHPMAILER

Il·lustració 15 : Logo PHPMailer. FONT: Externa[33]

VERSIÓ:6.0.5. La llibreria es troba en aquest GitHub →Enllaç.

PHP MAILER, és una llibreria desenvolupada per enviar de manera segura correus mitjançant PHP. Aquest no només permet enviar simples correus amb text simple, sinó que a més a més, permet enviar HTML, arxius adjunts... Cal destacar que és una de les formes més populars d'enviar correus mitjançant PHP.

Aquesta llibreria s'utilitza per enviar automàticament un correu a l'usuari després d'inscriure's a la cursa, registrar-se, restaurar la contrasenya o canviar l'adreça associada al compte.

ITEXTPDF

Il·lustració 16 : Logo ITextPDF. FONT: Externa[34]

VERSIÓ:5.5.10 (octubre 2016). La llibreria es troba en aquest repositori Maven.

ITEXTPDF, és una llibreria d'Android que permet la creació de PDF. Aquesta està en desenvolupament, actualment sobre la versió iText 7. A més a més de poder crear aquests documents utilitzant Java, permet convertir HTML a PDF, XML a PDF...

Es farà ús d'aquesta llibreria alhora de generar els documents PDF que contindran tota la informació del cotxe epsecificada en la pantalla del taller.

VOLLEY

Il·lustració 17 : Logo Volley. FONT: Externa[35]

VERSIÓ:1.1.0 (4 desembre, 2017). Llibreria en aquest GitHub.

VOLLEY, és una llibreria HTTP que permet comunicacions a través de la xarxa per aplicacions d'Android. Aquesta llibreria ofereix serveis com:

- Gestió planificada de les respostes de xarxa (*Shceduler*).
- Múltiples i concurrents connexions a la xarxa.
- Suport per establir prioritats en les peticions.
- Eines de *debuggeig* i seguiment.
- Permet cancel·lar peticions úniques o en bloc.
- Personalització de les sol·licituds.
- Permet rebre respostes a nivell de *string* ,imatges o amb el format JSON.
- Permet un emmagatzemament de respostes HTTP a la memòria cau de manera eficient i transparent.

Aquesta llibreria ens permet comunicar-nos a través de la xarxa, per tant, es podran realitzar peticions al *webservice* per obtenir informació de la base de dades, afegir o modificar-la .Caldrà declarar a l'arxiu *AndroidManifest*, el permís de connexió a Internet.

PASSWORD HASH i MYSQLi de PHP

Il·lustració 18 : Logo PHP. FONT: Externa[36]

PASSWORD HASH

Aquesta llibreria està **disponible a partir de la versió 5.3 de PHP**. Conté suport per molts dels algoritmes de *hashing*. Com per exemple l'algoritme: BCrypt, DES, Md5, PBKDF2, Sha-1,Sha-256, Sha-512...

Dins del projecte es fa ús d'aquestes dues funcions per **emmagatzemar contrasenyes:**

password_hash: Aquesta funció rep una contrasenya i un algoritme. D'algoritmes es poden destacar 2, PASSWORD_DEFAULT i PASSWORD_BCRYPT. El primer més segur, s'actualitza cada cop que apareix un algoritme més fort.
El segon, utilitzarà l'algoritme **CRYPT_BLOWFISH** per generar el *hash*. El resultat sempre serà un *String* de 60 caràcters.

Cal destacar que l'any 2011, es va descobrir un petit *bug* a la llibreria de PHP que afectava el 8è bit, el qual generava un caràcter que no es podia *manegar*. Actualment, els *hashes* generats per aquesta llibreria comencen per $2y$, la qual cosa, indica que ha estat generat per l'algoritme *debuggejat*. En cas que comencin per $2a$, significa que el *hash* va ser generat per l'algoritme antic.

password_verify: Aquesta funció permet obtenir *True/False* després de comprovar si la contrasenya coincideix amb un *hash* únic. Si coincideix, retorna *True*.

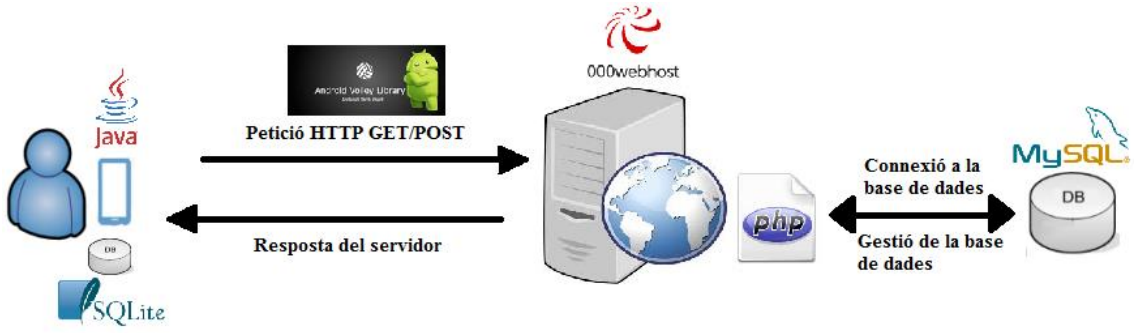
MYSQLi

Aquest mòdul està disponible a partir de les versions de **5+ de PHP**. L'extensió permet accedir a totes les funcionalitats que proporciona **MySQL 4.1** i posterior.

Aquesta extensió té un conjunt de mètodes que permeten gestionar les base de dades MySQL. Algunes de les funcions més utilitzades en els *scripts* de PHP del servidor són: **mysqli_affected_rows, mysqli_connect, mysqli_query...** Per més informació consultar l'ANNEX B. Estudi de les injeccions SQL

4.3 Arquitectura global de la comunicació

L'arquitectura del projecte serà **client-servidor** i tindrà la següent forma:



Il·lustració 19 : Arquitectura global comunicació. FONT: Pròpia

La part del **client** estarà composta per **l'aplicació d'Android** i una base de dades local per emmagatzemar les configuracions dels comandaments. El client és capaç de realitzar aquestes sol·licituds guiant l'usuari amb una interfície gràfica. Per tant, el **client** té un **paper actiu** en la comunicació.

Com a contrapartida, el **servidor es qui rep la petició**, espera que arribin per part dels clients, emparant un paper passiu.

L'aplicació es podrà comunicar amb el servidor mitjançant peticions HTTP tant GET/POST en funció de l'operació a realitzar . Aquestes peticions les gestionen petits *scripts* amb PHP, els quals, escaparan el valor de les variables i procediran a interactuar amb la base de dades. La base de dades remota permetrà enregistrar els nous *rallys*, tandes disponibles, peticions de seguretat, inscripcions i informació dels usuaris.

Els avantatges d'utilitzar aquesta arquitectura són les següents:

- **Centralització del control**: Els accessos, recursos i integritat de les dades dels usuaris són controlades pel servidor (exceptuant la configuració dels comandaments, es pot trobar més informació al disseny de la base de dades). Permet actualitzar dades u altres recursos.
- **Escalable:** Es pot augmentar la capacitat de clients i servidors per separat.
- **Fàcil manteniment:** Totes les modificacions com canvis de servidors no afectaran el client.

Desavantatges d'utilitzar aquesta arquitectura:

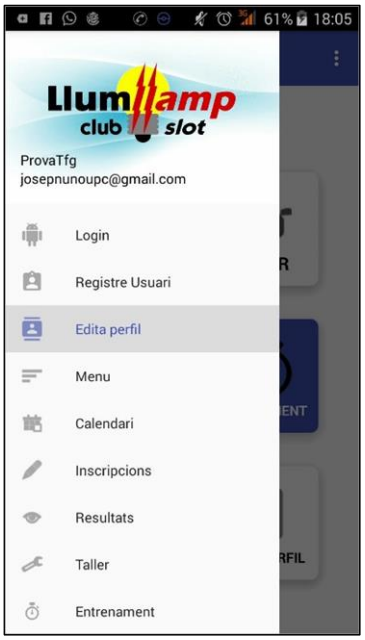
- **Congestió del tràfic:** A mesura que el nombre d'usuaris creix, gestionar moltes peticions pot posar en risc el funcionament correcte del servidor.
- **Falta de robustesa:** Si el servidor està caigut, els usuaris no podran rebre resposta a les seves peticions.
- **Ús de bon software i hardware:** Depenen del nombre d'usuaris i peticions que es reben, el hardware dels servidors queden obsolets reduint dràsticament el rendiment.

4.4 Arquitectura del *Software* de l'aplicació

En aquest subapartat del capítol s'explicaran els elements més importants de l'arquitectura de l'aplicació. Es tractaran punts claus alhora d'entendre com funciona el *navegation drawer*, les pantalles, la interfície gràfica i les classes.

4.4.1 *Navegation Drawer*

Un dels principals requisits era permetre a l'usuari desplaçar-se de manera senzilla i eficaç a través de l'aplicació. Per aquest motiu s'ha implementat el disseny d'un ***Navigation Drawer*** **(Menú desplegable)**. És un panell lateral que es desplaça des del costat esquerra de l'aplicació ocupant una part important de la pantalla. Aquest mostra una llistat d'elements i una capçelera.



Il·lustració 20 : Navigation Drawer. FONT: Pròpia

El *navigation drawer* és una vista que actua com a menú per canviar entre diferents fragments o pantalles de l'aplicació. Quan un usuari selecciona un element, aquest s'oculta automàticament deixant pas al fragment que s'estava sobreposant. Per tant, **el patró de comportament principal de l'aplicació és Navigation Drawer+ Fragment.**

Per acabar d'entendre com funciona un fragment i la seva estructura, revisar l'apartat de l'Annex A. Aquí s'explica detalladament com funcionen activitats i fragments.

4.4.2 Pantalles de l'aplicació

El disseny de pantalles permet veure de manera global, com es desplaçarà l'usuari dins de l'aplicació. També, es podrà conèixer quines de les pantalles sol·liciten informació al servidor i quines a la base de dades local.

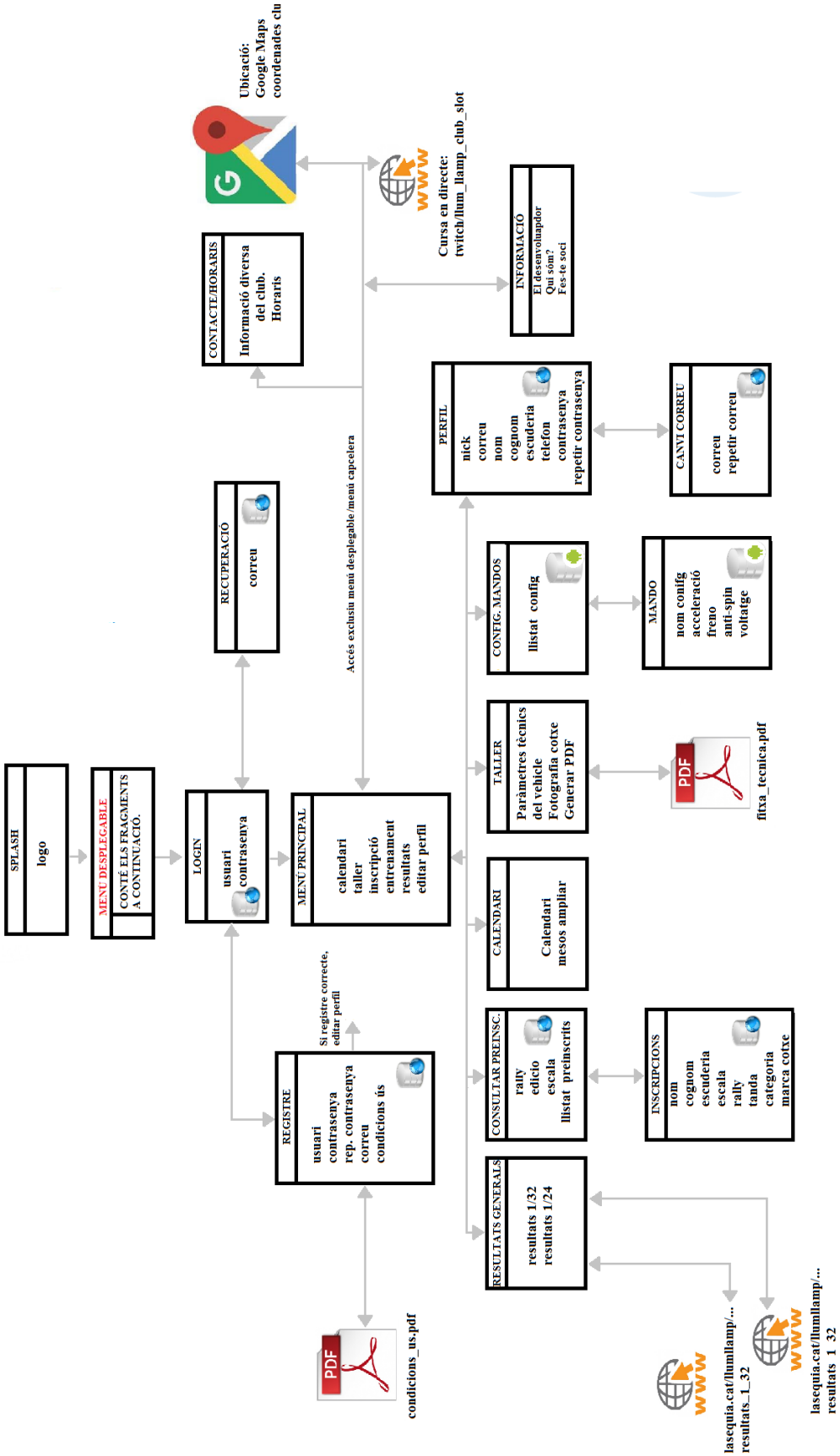
Cal destacar que el ***NavegationDrawer*** **(MenuDesplegable)** es converteix en l'activitat principal després del *splash screen,* totes les vistes a continuació són fragments continguts dintre seu i que ocupen l'àrea central d'aquesta activitat.

La comunicació entre pantalles està pensada i enfocada a l'ús del menú principal com una espècie de *home.* L'usuari l'utilitzarà per accedir a les funcionalitats principals i acabarà retornant un cop premi el botó *back()* del terminal Android.

Quan un usuari premi el botó d'enrere sobre el menú principal, apareixerà un *dialog* on es preguntarà si realment vol sortir de l'aplicació.

Il·lustració 21 : Diàleg del menú principal. FONT: Pròpia

A la següent pàgina es podrà veure el conjunt de pantalles que formen íntegrament l'aplicació.



Il·lustració 22 : Disseny de pantalles. FONT: Pròpia

4.4.3 La interfície gràfica

La interfície està dissenyada amb el propòsit de ser simple i intuïtiva. D'una manera senzilla, l'usuari podrà navegar per ella sense cap dificultat i amb poc ús comprendre com funciona. A més a més, ha de ser capaç de resoldre tot el catàleg de requisits que es va proposar inicialment.

Per poder veure el disseny de cadascuna de les pantalles, es poen trobar en l'Annex C. Manual de l'usuari

El disseny de la interfície gràfica està principalment pensat per a dispositius mòbils amb **pantalles de 5 polzades amb una resolució de xxhdpi 1080x1920 o no superiors a 420dpi**. Tanmateix, per a pantalles més petites, s'ha implementat un *scrollview*, d'aquesta manera, permet desplaçar-se a munt i avall com si es tractes d'un navegador, com es pot comprovar en el següent exemple:

Il·lustració 23 : Exemple de Scroll View. FONT: Pròpia

Cal destacar però, que per a pantalles amb una densitat de resolució alta (pantalla més gran) el resultat és una mica pobre, ja que les imatges no estan escalades amb diferents resolucions. Per tant, com a millora de futur, seria interessant ajustar tot el contingut a les principals resolucions dels dispositius actuals.

4.4.4 Classes de l'aplicació

A continuació es farà una ressenya extensa sobre les classes que s'han creat per l'aplicació. Aquestes es poden dividir en dos grups: classes Java exclusives per gestionar la interfície gràfica i altres que tenen una funció determinada.

Classes Java per la interfície gràfica:

Totes les pantalles que poden ser activitats o fragments, tenen associat un *layout* en forma de fitxer XML que conté la informació sobre els diferents elements de la IU utilitzats.

Cada *layout* té assignada una Classe Java que permet interactuar i realitzar les accions necessàries per cada pantalla. En aquestes classes Java serà el lloc on es posaran els *listeners* pels elements de la IU, es comprovaran els paràmetres introduïts, realitzaran les peticions al servidor corresponents...

Nom de la pantalla	Classe .Java	Layout .XML
Calendari	Fragment_calendari	Fragment_calendari
Canvi de correu	Fragment_canvi_Correu	Fragment_canvi_Correu
Vies de contacte	Fragment_contacte	Fragment_contacte
Gestió del cotxe	Fragment_gestio_cotxe	Fragment_gestio_cotxe
Informació	Fragment_informacio	Fragment_informacio
Inscripcions	Fragment_inscripcions	Fragment_inscripcions
Consultar preinscripció	Fragments_inscripcions_database	Fragments_inscripcions_database
Login	Fragment_login	Fragment_login
Gestió del mando	Fragment_mandos	Fragment_mandos
Configuracions Mando	Fragment_mandos_database	Fragment_mandos_database
Menú Principal	Fragment_menu	Fragment_menu
Editar Perfil	Fragment_perfil	Fragment_perfil
Recuperació del compte	Fragment_recuperacio	Fragment_recuperacio
Registre d'un usuari nou	Fragment_registre	Fragment_registre
Resultats generals	Fragment_results	Fragment_results
-----	SplashScreen	Splash_screen
-----	MenuDesplegable	Activity_menu_desplegable

Taula 4 : Disseny de classes i layouts. FONT: Pròpia

Altres classes Java:

Control_base_dades: Aquesta classe pública hereta funcions (*extends*) de la classe abstracte SQLITEOPENHELPER, la qual, ens permet gestionar les bases de dades en l'àmbit d'Android. Quan s'utilitza una base de dades a Android, es crea una subclasse ajustada d'aquesta. Les principals funcions a destacar són:

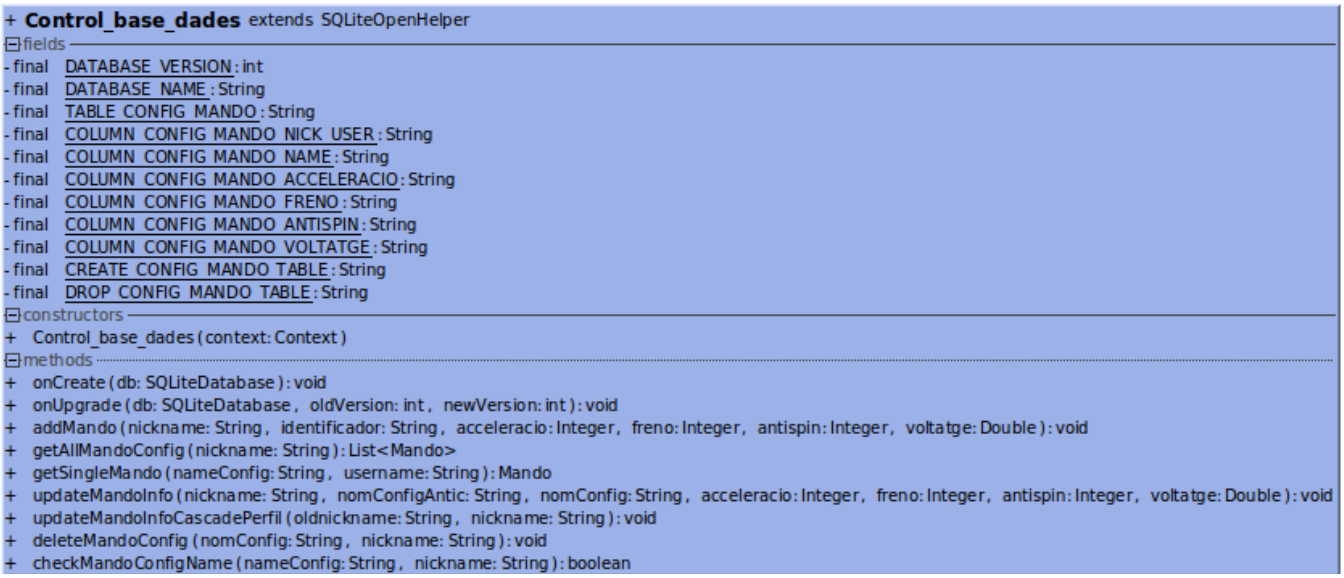
SQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int Version), constructor públic de l'objecte que ens ajudarà a obrir, crear o manegar la base de dades.

OnCreate(SQLiteDatabase db), com indica el seu nom, només s'utilitza en el moment de creació de la base de dades. S'executa de manera única quan s'instal·la l'aplicació, normalment composta per sentències SQL natiu.

OnUpgrade(SQLiteDatabase db, int oldVersion, int newVersion), cada cop que la base de dades necessita actualitzar-se es fa ús d'aquesta funció. Aquest mètode s'executa dins d'una transacció, si es produeix algun error que dispari una excepció, automàticament tots els canvis es desfaran *(rollback())*

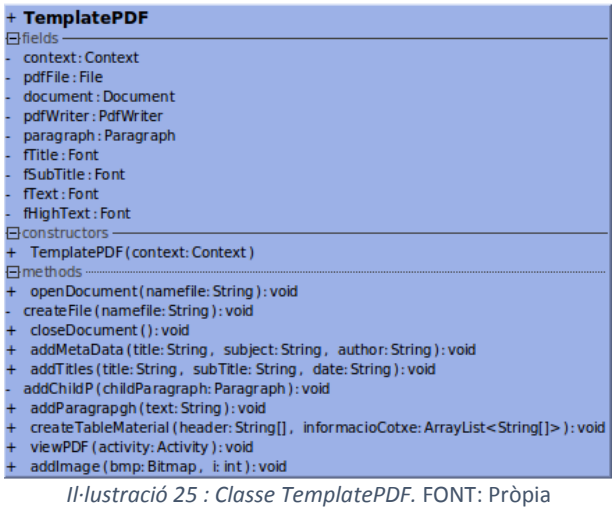
Quan es realitzen consultes a la base de dades, s'utilitza la funció **Cursor**, que permet accedir al resultat d'una consulta de manera fàcil, ja que disposa de mètodes públics com podrien ser *getColumnNames(), getColumnIndex*... que retornen els resultats de les consultes en forma d'*array*.

A més a més de les funcions esmenades, s'han afegit funcions noves totalment ajustades per gestionar la base de dades encarregada d'emmagatzemar les configuracions dels comandaments del pilot.



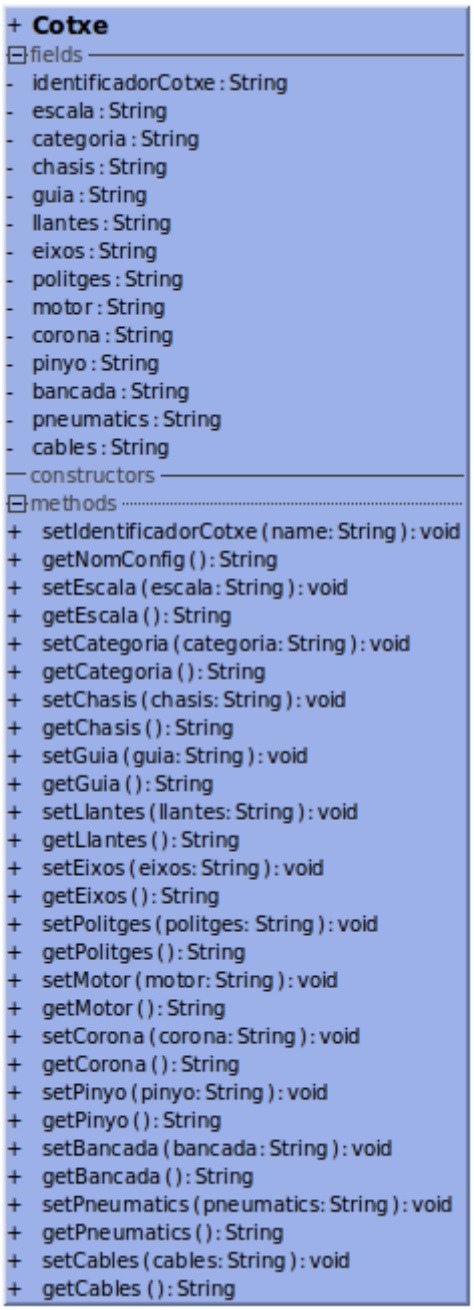
Il·lustració 24 : Classe Control_base_dades. FONT: Pròpia

TemplatePDF: Classe que s'utilitza al fragment gestió_cotxe. La seva principal funcionalitat és establir una **plantilla única** per a la creació de tots els PDF de l'aplicació. És on es defineix el tipus de font, la mida dels títols o paràgrafs...Fa ús de mètodes propis que permeten afegir imatges, paràgrafs, taules, etc... Als documents. Aquests mètodes fan ús de les funcions que proporciona ***iTEXTPDF***.



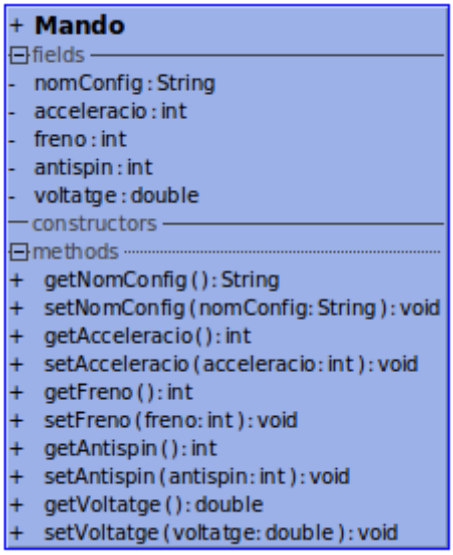
Il·lustració 25 : Classe TemplatePDF. FONT: Pròpia

Cotxe: Classe pública utilitzada en el fragment gestió_cotxe, aquesta serveix per emmagatzemar tota la informació del vehicle i poder accedir-hi fàcilment. Té mètodes propis per tal d'accedir als atributs privats com per exemple, assignar la marca de les politges, corones, xassís, carrosseries...



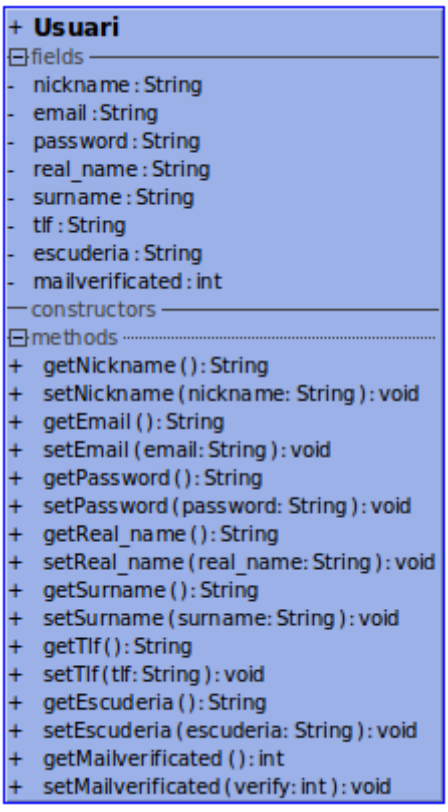
Il·lustració 26 : Classe Cotxe. FONT: Pròpia

Mando: Classe pública utilitzada en el fragment mandos, aquesta serveix per emmagatzemar tota la informació dels comandaments i poder obtenir-la fàcilment. També, quan es realitzen consultes a la base de dades, serveix com a recipient dels valors obtinguts en la consulta, és a dir, ens permet gestionar la informació de manera senzilla i unificada en forma d'objecte.



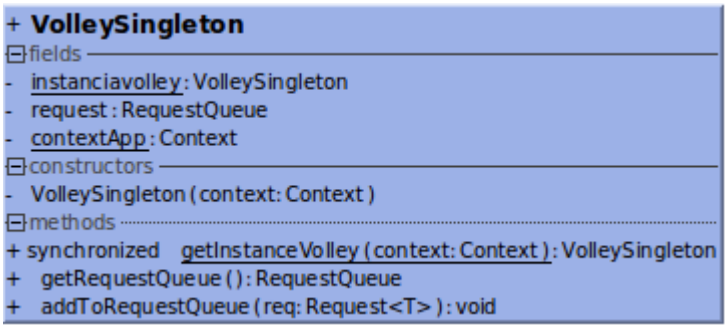
Il·lustració 27 : Classe Mando. FONT: Pròpia

Usuari: La importància d'aquesta classe pública resideix en emmagatzemar les dades obtingudes amb la consulta a la base de dades remota. S'utilitza com a recipient on es pot consultar les credencials de l'usuari sense haver de fer una consulta explícita al servidor.



Il·lustració 28 : Classe Usuari. FONT: Pròpia

VolleySingleton: Quan es fa ús de la llibreria Volley, la documentació oficial recomana encapsular totes les peticions fent servir un *Singleton*. La nostra aplicació utilitza bastant la xarxa, pel que és més eficient tenir una classe ***Singleton***. També és recomanable perquè el *RequestQueue* generat a l'enviar una petició, tindrà un cicle de vida que serà el de l'aplicació, mentre que si no s'implementés així, només tindria un cicle de vida limitat, depèn del context en què es trobés.



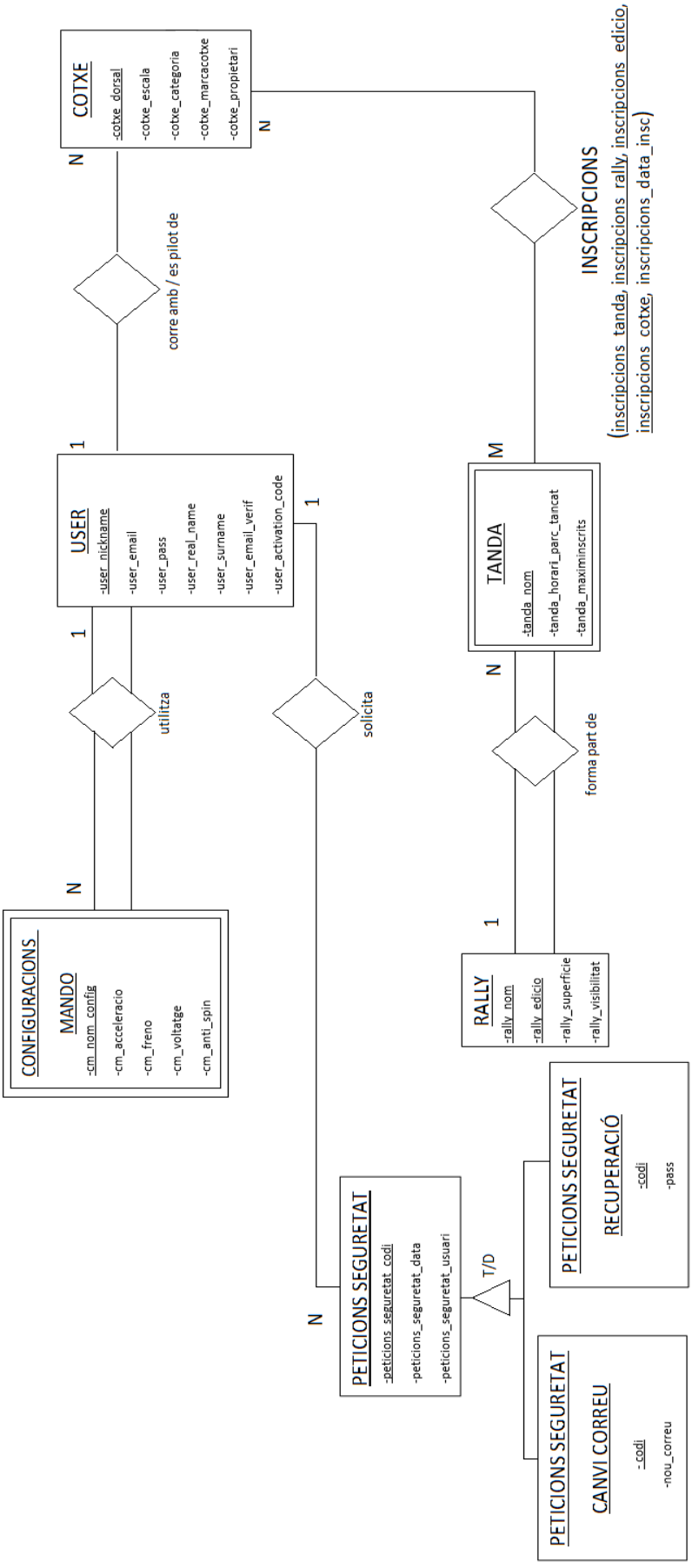
Il·lustració 29 : Classe VolleySingleton. FONT: Pròpia

4.5 Disseny de la base de dades

La base de dades ha de ser capaç de gestionar la informació necessària relacionada amb els comptes d'usuari, els *rallys*, les tandes, peticions de seguretat, configuracions del comandament i a més a més, les inscripcions a les curses.

Respecte a les **peticions de seguretat**, s'entén com a sol·licituds que posen en compromís la seguretat del compte si s'accepten. Per exemple, canviar la direcció del correu per un altre suposa un canvi important. De la mateixa manera, el procés de restauració, on es dóna una nova contrasenya a l'usuari si perd les seves credencials.

A la següent pàgina es podrà veure el diagrama E/R proposat i a continuació, el codi SQL constructor de la base de dades.



Il·lustració 30 : Disseny de la base de dades. FONT: Pròpia

El codi constructor SQL encarregat de crear les taules de la base de dades és el següent:

```
CREATE TABLE `user` (
  `user_nickname` varchar(20) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `user_email` varchar(60) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `user_pass` varchar(100) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `user_real_name` varchar(20) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  `user_surname` varchar(20) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  `user_tlf` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  `user_escuderia` varchar(25) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  `user_mail_verif` int(1) DEFAULT '0',
  `user_activation_code` varchar(100) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`user_nickname`),
  UNIQUE KEY `user_email` (`user_email`),
  UNIQUE KEY `user_activation_codi` (`user_activation_code`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Il·lustració 31 : SQL taula user. FONT: Pròpia

```
CREATE TABLE `cotxe` (
  `cotxe_dorsal` int(8) NOT NULL AUTO_INCREMENT,
  `cotxe_escala` varchar(5) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `cotxe_categoria` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `cotxe_marcacotxe` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  `cotxe_propietari` varchar(20) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`cotxe_dorsal`),
  KEY `cotxe_propietari` (`cotxe_propietari`),
  CONSTRAINT `cotxe ibfk 1` FOREIGN KEY (`cotxe_propietari`) REFERENCES `user` (`user_nickname`)
  ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8
```

Il·lustració 32 : SQL taula cotxe. FONT: Pròpia

```
CREATE TABLE `peticions_seguretat` (
  `peticions_seguretat_codi` varchar(255) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `peticions_seguretat_data` date DEFAULT NULL,
  `peticions_seguretat_usuari` varchar(20) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  PRIMARY KEY (`peticions_seguretat_codi`),
  KEY `peticions_seguretat_usuari` (`peticions_seguretat_usuari`),
  CONSTRAINT `peticions seguretat ibfk_1` FOREIGN KEY (`peticions_seguretat_usuari`)
  REFERENCES `user` (`user_nickname`)
  ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Il·lustració 33 : SQL taula peticions_seguretat. FONT: Pròpia

```
CREATE TABLE `peticions_seguretat_canvi_correu` (
  `codi` varchar(255) COLLATE utf8_spanish_ci NOT NULL,
  `nou_correu` varchar(60) COLLATE utf8_spanish_ci NOT NULL,
  PRIMARY KEY (`codi`),
  CONSTRAINT `peticions_seguretat_canvi_correu_ibfk_1` FOREIGN KEY (`codi`)
  REFERENCES `peticions_seguretat` (`peticions_seguretat_codi`)
  ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci
```

Il·lustració 34 : SQL taula peticions_seguretat_canvi_correu. FONT: Pròpia

```
CREATE TABLE `peticions_seguretat_recuperacio` (
  `codi` varchar(255) COLLATE utf8_spanish_ci NOT NULL,
  `pass` varchar(100) COLLATE utf8_spanish_ci NOT NULL,
  PRIMARY KEY (`codi`),
  CONSTRAINT `peticions_seguretat_recuperacio_ibfk_1` FOREIGN KEY (`codi`)
  REFERENCES `peticions_seguretat` (`peticions_seguretat_codi`)
 ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_spanish_ci
```

Il·lustració 35 : SQL codi peticions_seguretat_recuperacio. FONT: Pròpia

```
CREATE TABLE `rally` (
  `rally_nom` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `rally_edicio` int(11) NOT NULL,
  `rally_superficie` varchar(10) CHARACTER SET utf8 COLLATE utf8_spanish_ci  DEFAULT NULL,
  `rally_visibilitat` varchar(10) CHARACTER SET utf8 COLLATE utf8_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`rally_nom`,`rally_edicio`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Il·lustració 36 : SQL codi rally. FONT: Pròpia

```
CREATE TABLE `tanda` (
  `tanda_nom` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `tanda_rally` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `tanda_edicio` int(11) NOT NULL,
  `tanda_horari_parc_tancat` int(11) DEFAULT NULL,
  `tanda_maximinscrits` int(11) DEFAULT NULL,
  PRIMARY KEY (`tanda_nom`,`tanda_rally`,`tanda_edicio`),
  KEY `tanda_rally` (`tanda_rally`,`tanda_edicio`),
  CONSTRAINT `tanda_ibfk_1` FOREIGN KEY (`tanda_rally`, `tanda_edicio`)
  REFERENCES `rally` (`rally_nom`, `rally_edicio`)
  ON DELETE CASCADE ON UPDATE CASCADE
 ) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Il·lustració 37 : SQL codi tanda. FONT: Pròpia

```
CREATE TABLE `inscripcions` (
  `inscripcions_tanda` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `inscripcions_rally` varchar(15) CHARACTER SET utf8 COLLATE utf8_spanish_ci NOT NULL,
  `inscripcions_edicio` int(11) NOT NULL,
  `inscripcions_cotxe` int(8) NOT NULL,
  `inscripcions_data_insc` date DEFAULT NULL,
  PRIMARY KEY (`inscripcions_rally`,`inscripcions_cotxe`,`inscripcions_tanda`,`inscripcions_edicio`),
  KEY `inscripcions_tanda` (`inscripcions_tanda`,`inscripcions_rally`,`inscripcions_edicio`),
  KEY `inscripcions_cotxe` (`inscripcions_cotxe`),
  CONSTRAINT `inscripcions_ibfk_1` FOREIGN KEY (`inscripcions_tanda`, `inscripcions_rally`, `inscripcions_edicio`)
 REFERENCES `tanda` (`tanda_nom`, `tanda_rally`, `tanda_edicio`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `inscripcions_ibfk_2` FOREIGN KEY (`inscripcions_cotxe`) REFERENCES `cotxe` (`cotxe_dorsal`)
  ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Il·lustració 38 : SQL codi inscripcions. FONT: Pròpia

Per últim, la única taula gestionada de manera local al dispositiu de l'usuari:

```
private final String CREATE_CONFIG_MANDO_TABLE = "CREATE TABLE configuracions_mando(" +
        "cm_nom_config varchar(30) NOT NULL," +
        "cm_acceleracio INTEGER," +
        "cm_freno INTEGER," +
        "cm_anti_spin INTEGER," +
        "cm_voltatge DOUBLE," +
        "cm_user_mando varchar(20) NOT NULL," +
        "PRIMARY KEY (cm_nom_config,cm_user_mando))";
```

Il·lustració 39 : SQL codi taula config. Mando. FONT: Pròpia

4.6 Disseny de la recepció de peticions al servidor

Per a ser possible la gestió de la base de dades en el servidor, s'utilitzen diferents *scripts* PHP per tal de rebre les peticions i retornar una resposta a l'usuari. Aquests fitxers retornaran un codi diferent en funció de si l'operació s'ha pogut realitzar correctament, o pel contrari, ha sorgit un error o impediment.

Els fitxers encarregats de rebre peticions de l'aplicació retornaran la informació necessària amb **format JSON.**

Pantalla	Peticions HTTP a:	Tipus	Descripció breu
Login	ConsultaUsuariLogin.php	**POST**	Permet iniciar sessió a l'usuari.
Registre	Registre_Usuari.php	**POST**	Sol·licita el registre d'un nou usuari a la BD.
Recuperació	EnviaInfoAccount.php	**POST**	Envia un correu a l'e-mail per recuperar la informació del compte i generar nova contrasenya provisional.
Perfil	Actualitza_pass.php	**POST**	Canvia la contrasenya per una de nova.
	Actualitza_Usuari.php	**POST**	Actualitza dades del compte i el nick.
	ConsultaUsuari.php	**GET**	Obté la informació del compte.
Canvi Correu	EnviaCanviCorreu.php	**POST**	Envia una confirmació al correu del compte per canviar la direcció de correu.
Inscripcions_data	ConsultaInscrits.php	**GET**	Consultar inscrits al rally.
	ConsultaRallys.php	**GET**	Obté els rallys disponibles actualment.
Inscripcions	Registre_Cotxe.php	**POST**	Emmagatzema un cotxe i li assigna un nombre de dorsal.
	Registra_Inscripcio.php	**POST**	Genera una inscripció nova per l'usuari.
	ConsultaEmailConfirmat.php	**GET**	Verifica si l'usuari que vol inscriure's té el correu confirmat.
	ConsultaTandaRallys	**GET**	Obté les tandes disponibles pel rally seleccionat.
	ConsultaRallys	**GET**	Obté els rallys disponibles actuals.

Taula 5 : Recepció de peticions I. FONT: Pròpia

Els fitxers encarregats de rebre les peticions provinents del correu són els següents:

Servei de Correu	Activation.php	**GET**	Activa el compte d'usuari.
	Confirm_new_email.php	**GET**	Confirma el canvi de correu del compte.
	Generate_new_pass.php	**GET**	Genera una nova contrasenya per l'usuari.

Taula 6 : Recepció de peticions II. FONT: Pròpia

4.7 Disseny de la seguretat

Un dels objectius més importants és que l'aplicació sigui el màxim de segura possible. Per aconseguir això s'han pres les següents mesures.

- Les dades relacionades amb **contrasenyes de l'usuari estan encriptades** amb l'algoritme BCrypt. D'aquesta manera s'evita que qualsevol persona que gestioni la base de dades pugui obtenir-les.
- S'ha controlat les **injeccions SQL** que pugues realitzar un usuari malintencionat mitjançant l'escapament de variables. També es comproven tots els paràmetres que s'introdueix a l'aplicació. S'ha realitzat un estudi exhaustiu d'injeccions SQL que es pot trobar en l'Annex B. Estudi de les injeccions SQL.
- El compte de l'usuari es pot **restaurar en cas de robatori** i s'ha de **confirmar que es vol canviar la direcció de correu**. Per veure el procés que s'ha de seguir, consultar l'apartat 6. Proves del sistema.
- Quan algú vol enviar una petició a qualsevol document que es troba dins del directori /bd_slot_club/ publicat a 000webhost, necessita una **autentificació**. En cas del directori /bd_slot_club_email/ que s'utilitza per rebre les peticions provinents del correu, es necessari **un codi únic de mida variable** per tal de processar i fer efectiva la sol·licitud d'activació de compte, canvi de correu o recuperació.
- Es **permet** l'inici de vàries **sessions no concurrents des de diferents dispositius alhora**. Tot i així, per exemple, si l'usuari 1 modifica el correu o el *nickname* del compte, quan l'usuari N intenti sol·licitar qualsevol tipus de petició com una inscripció o editar el seu perfil, se li tancarà sessió i se l'informarà que algú ha modificat les seves dades. Per consegüent, l'usuari N haurà d'iniciar un procés de restauració si no havia sigut ell.
- Actualment com a primera versió de l'aplicació s'utilitza la llibreria **Volley**, la qual permet enviar sol·licituds **HTTP**, per tant, seria interessant afegir un certificat SSL específic del servei web definitiu. D'aquesta manera les sol·licituds serien totalment segures i certificades.

5. Implementació

En aquest capítol de la memòria es tractarà la manera d'abordar les principals fases del projecte. Es mostrarà petites parts del codi capaces d'il·lustrar com s'ha implementat les diverses funcionalitats.

S'explicarà com s'interactua amb els elements de la IU, la forma amb la qual l'aplicació es comunica amb el servei web, com els *scripts* PHP són capaços de gestionar la base de dades, l'automatisme d'enviar correus amb PHPMailer entre altres apartats importants.

5.1 Elements de l'aplicació Android

Dins d'aquest apartat es tractaran els elements més rellevants de la implementació del disseny sobre l'aplicació Android.

S'explicarà quin és el paper de l'arxiu manifest, l'arxiu build.gradle i del directori res.

També es mostrarà com s'ha implementat la comunicació amb Volley, com interactuen els elements de la interfície gràfica, com es gestionen les bases de dades, la importància de les tasques asíncrones i per últim com es genera el documentPDF amb iTEXTPDF.

5.1.1 L'arxiu manifest

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Il·lustració 40 : Codi arxiu manifest. FONT: Pròpia

L'arxiu AndroidManifest.xml és un arxiu que es troba en el directori arrel. Serà el lloc on es declararan els permisos de l'aplicació:

- **Camara**: L'usuari pot utilitzar la càmera per fotografiar el seu cotxe d'Slot en el fragment gestió_cotxe.
- **Internet**: S'utilitza en la comunicació entre l'App i la base de dades remota.
- **Write / Read_external_storage**: L'utilitza el fragment gestió del cotxe quan vol llegir el contingut del PDF generat i també, alhora d'emmagatzemar el document en la memòria del dispositiu.

A més a més en aquest document, s'especifica amb l'atribut ***intent-filter*** quina serà la *splash screen* (activitat que apareix inicialment durant 3 segons amb el logotip de Llum Llamp) un cop s'obri l'aplicació.

5.1.2 L'arxiu build.gradle

Gradle és un sistema de compilació que reuneix en un sol sistema, les millors prestacions d'altres sistemes de compilació. Està basat en **JVM** (*Java Virtual Machine*). Gradle és un *plugin* que facilita l'actualització i exportació d'un projecte a un altre.

Dins d'aquest arxiu trobem una part molt important del projecte. S'especifica paràmetres com: quina versió d'Android es necessària, les llibreries o dependències, identificador del projecte, *debuggejador* utilitzat, versió del codi...

La versió **mínima SDK és la 15** i la **màxima 26**. La mínima correspon a Android 4.0.3 (***Ice_cream_sandwitch_mr1***), una versió millorada de la ***4.0 Ice_cream_sandwitch i anterior a Jelly_bean (4.1)***. Per tant, qualsevol persona que tingui una terminal Android amb S.O posterior a desembre de 2011, podrà utilitzar l'aplicació, és a dir, segons l'estudi de muycomputer, el **99% dels usuaris d'Android.**

Dependències de l'aplicació:

```
implementation fileTree(include: ['*.jar'], dir: 'libs')
implementation 'com.android.support:appcompat-v7:26.1.0'
implementation 'com.android.support.constraint:constraint-layout:1.0.2'
implementation 'com.android.support:design:26.1.0'
testImplementation 'junit:junit:4.12'
androidTestImplementation 'com.android.support.test:runner:1.0.1'
androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
implementation 'com.android.support:cardview-v7:26.1.0'
implementation 'com.itextpdf:itextg:5.5.10'
implementation 'com.android.volley:volley:1.1.0'
implementation 'com.android.support:support-v4:26.1.0'
```

Il·lustració 41 : Codi dependències. FONT: Pròpia

El paquet **de suport de compatibilitat** d'Android conté diverses llibreries que es poden incloure a l'aplicació. Aquestes no venen integrades directament en el *framework*. S'utilitzen per oferir funcionalitats noves compatibles amb les versions anteriors, proporcionant diferents elements IU que en antany no es trobaven disponibles, però, en l'actualitat si. Cal destacar la llibreria d'**android.support-V4**, principalment utilitzada per la gestió dels fragments i el *NavegationDrawer*. **L'android.support-v7** per a poder utilitzar l'element visual del *cardView* en versions anteriors.

5.1.3 El directori res

Aquest directori serà l'encarregat d'emmagatzemar els recursos relacionats amb el disseny gràfic. Es pot dividir en 4 parts:

- ***drawable*:** Dins d'aquest directori trobem TOT el contingut gràfic que utilitza l'aplicació. No es poden afegir carpetes per classificar el contingut del directori perquè el compilador no els té en compte i els exclou, per aquest motiu, la solució proposada per la documentació oficial és afegir un prefix a cada imatge. Per exemple:
  lm_citroen.gif* → *logomarca_NomMarca. extensió
- ***mipmap*:** Contindrà el logotip de l'aplicació.
- **menú:** Directori generat automàticament a l'importar el *navegationDrawer*. Conté el seu document XML i les opcions.
- ***values*:** Es pot trobar el contingut dels *spinners* principalment. A més a més cal destacar que el document ***STRINGS.XML,*** el qual, conté TOTS els *strings* que formen part dels documents XML de l'aplicació. D'aquesta manera, es pot modificar l'idioma fàcilment afegint nous *Strings*.

5.1.4 Interacció amb els elements de la interfície

El contingut de les classes Java exclusives per gestionar la interfície gràfica serà el lloc on es podrà trobar funcions d'aquest estil.

```
Button bt_recuperacio;
bt_recuperacio=actualview.findViewById(R.id.bt_recuperacio_envia);
bt_recuperacio.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String getEditextCorreu=recuperacio_correu.getText().toString();
        if (checkCorreu(getEditextCorreu)&&isEmailPattern(getEditextCorreu))
        new TaskRecuperaCompte().execute(getEditextCorreu);
    }
});
```

Il·lustració 42 : Codi interacció IU. FONT: Pròpia

Per poder interactuar amb un botó, primer s'ha de declarar un Objecte Java del tipus *Button*. Posteriorment, mitjançant la funció *findViewByID()* es pot associar l'objecte Java i la seva representació gràfica dins del layout XML.

Aquest fragment de codi pertany a la classe Java fragment_recuperació i permet veure quina depuració dels *inputs* es duu a terme. Per comprovar quan algú ha fet click sobre aquest, s'ha d'implementar un *listener* i dins d'aquest, fer les accions necessàries.

Concretament, en aquest exemple, s'obté el correu especificat dins d'un *editext* i es comprova que tingui la longitud correcta i compleixi el patró d'un correu, utilitzant la funció *isEmailPatter().*

```
private boolean isEmailPattern(String email){

    Pattern pattern = Patterns.EMAIL_ADDRESS;
    if (pattern.matcher(email).matches())
        return true;
    else{
       Toast.makeText(getActivity(), "Introdueix un correu và-
       lid!", Toast.LENGTH_LONG).show();
        return false;
    }
}
```

Il·lustració 43 : Codi isEmailPattern. FONT: Pròpia

Per a realitzar la comprovació es fa ús de l'objecte Pattern, el qual permet, comprovar mitjançant el Patterns.EMAIL_ADRRESS si el *String* té la forma corresponent a un correu.
Posteriorment, un cop realitzada la comprovació, s'inicia una tasca asíncrona.

5.1.5 Gestió de la base de dades local

La base de dades local dins del terminal Android està formada principalment per una sola taula que té la següent forma:

```
private final String CREATE_CONFIG_MANDO_TABLE = "CREATE TABLE configuracions_mando(" +
        "cm_nom_config varchar(30) NOT NULL," +
        "cm_acceleracio INTEGER," +
        "cm_freno INTEGER," +
        "cm_anti_spin INTEGER," +
        "cm_voltatge DOUBLE," +
        "cm_user_mando varchar(20) NOT NULL," +
        "PRIMARY KEY (cm_nom_config,cm_user_mando))";
```

Il·lustració 44 : Gestió de la base de dades local. FONT: Pròpia

La classe Control_base_dades disposa de les funcions necessàries per gestionar la base de dades de manera local, aquesta és un exemple:

```
public Mando getSingleMando(String nameConfig,String username) {
    Mando mando = new Mando();
    String[] columns = {
            COLUMN_CONFIG_MANDO_NAME,
            COLUMN_CONFIG_MANDO_ACCELERACIO,
            COLUMN_CONFIG_MANDO_FRENO,
            COLUMN_CONFIG_MANDO_ANTISPIN,
            COLUMN_CONFIG_MANDO_VOLTATGE,
            COLUMN_CONFIG_MANDO_NICK_USER};
    SQLiteDatabase db = this.getReadableDatabase();

    String selection = COLUMN_CONFIG_MANDO_NAME + " = ?"+ " AND " + COLUMN_CONFIG_MANDO_NICK_USER + " = ?";
    String[] selectionArgs = {nameConfig, username};
    Cursor cursor = db.query(TABLE_CONFIG_MANDO, //Table to query
            columns,                    //columns to return
            selection,                  //columns for the WHERE clause
            selectionArgs,              //The values for the WHERE clause
             groupBy: null,                    //group the rows
             having: null,                   //filter by row groups
             orderBy: null);                   //The sort order
    if (cursor.moveToFirst()) {
        do {
            mando.setNomConfig(cursor.getString(cursor.getColumnIndex(COLUMN_CONFIG_MANDO_NAME)));
            mando.setAcceleracio(cursor.getInt(cursor.getColumnIndex(COLUMN_CONFIG_MANDO_ACCELERACIO)));
            mando.setFreno(cursor.getInt(cursor.getColumnIndex(COLUMN_CONFIG_MANDO_FRENO)));
            mando.setAntispin(cursor.getInt(cursor.getColumnIndex(COLUMN_CONFIG_MANDO_ANTISPIN)));
            mando.setVoltatge(cursor.getDouble(cursor.getColumnIndex(COLUMN_CONFIG_MANDO_VOLTATGE)));
        } while (cursor.moveToNext());
    }
    cursor.close();
    db.close();
    return mando;
}
```

Il·lustració 45 : Codi getSingleMando. FONT: Pròpia

La funció ***getSingleMando*** s'utilitza per obtenir tota la informació relacionada amb els paràmetres de la configuració del comandament. A aquesta funció se li ha de passar el nom de la configuració introduït per l'usuari.

Els ***strings*** que es passen com a paràmetre estan **perfectament escapats ja que s'utilitzen sentències parametritzades, d'aquesta manera, s'eviten injeccions SQL.** El segon l'obté de la variable global **logged_user** encarregada d'emmagatzemar la informació de l'usuari.

Per realitzar la consulta s'utilitza la funció ***query***, disponible gràcies a hereta els mètodes de la classe SQLiteOpenHelper. Aquesta funció treballa amb 7 paràmetres:
-***table*:** Nom de la taula a consultar.
-***columns*:** Quines columnes s'han de retornar en la consulta.
-***selectionArgs***: Columnes a consultar en el *select*. És una sentència SQL del tipus Where.
-***groupBy:*** Declarar com s'agrupen les columnes. Si es passa com a paràmetre un *null*, les columnes no s'agruparan.
-***having:*** Té la mateixa estructura que una sentència SQL Having. Serveix per indicar quin grup de columnes s'inclourà en el cursor.
-***orderBy****:* Com s'ordenen les columnes. Mateixa estructura que la sentencia SQL OrderBy
-***limit****:* Nombre de columnes que ha de retornar com a màxim la sentència a executar.

Per obtenir tota la informació retornada per part de la consulta a la base de dades és imprescindible fer ús de la classe mando. D'aquesta manera es pot encapsular el resultat dins d'un objecte fàcil de gestionar.

5.1.6 La comunicació amb Volley

La llibreria que permet enviar sol·licituds HTTP entre el dispositiu client al servidor i viceversa és la llibreria Volley. Per poder **funcionar correctament**, és imprescindible afegir **permisos de connexió a xarxa** en **l'arxiu manifest**.

Sol·licitud POST:

Per enviar exitosament la sol·licitud POST al servidor, serà necessari la implementació de 4 mètodes:

1. **onResponse.**
2. **onErrorResponse.**
3. **getParams.**
4. **getHeaders.**

A continuació es mostrarà un exemple de cadascun d'aquests dins de l'aplicació i les principals característiques.

Inicialització d'una petició:

```
String base_dades_remota=getString(R.string. ip_000webhost);
String URL = base_dades_remota + "/ConsultaUsuariLogin.php";
URL = URL.replace(" ", "%20");
StringRequest stringRequest = new StringRequest(Request.Method.POST,
URL, ...) {...}
```

Il·lustració 46 : Codi Volley I. FONT: Pròpia

Al paràmetre de l'URL s'especificarà quin és la direcció on es realitzarà la petició. Per tal de codificar els espais que es puguin troben a l'URL, es substitueixen pel codi %20.

```
@Override
public void onResponse(String response) {
       JSONObject jsonObject = new JSONObject(response);
       JSONArray json = jsonObject.optJSONArray("user");
       jsonObject = json.getJSONObject(0); ...}
```

Il·lustració 47 : Codi Volley II. FONT: Pròpia

Per obtenir la informació del servidor, s'ha d'implementar un mètode *listener* per "escoltar" la resposta:

Com que la petició és del tipus *StringRequest* , Volley espera rebre una resposta amb format de *String* , tot i així, sempre serà en format JSON. Per aquest motiu, la resposta pot convertir-se i tractar-se com un objecte JSON. La ***funció jsonObject.optJSONArray*** permet extreure tota la informació que es trobi dintre del node "user", transformant-lo en un *array*.

No sempre les respostes provinents del servidor són resultat d'una petició exitosa, en diversos casos, podem rebre errors, per tant, també serà necessari implementar un mètode que sigui capaç d'identificar-los.

```
@Override
public void onErrorResponse(VolleyError error) {
    Toast.makeText(getActivity(), "No s'ha pogut connectar amb el servidor, torni a intentar-ho", Toast.LENGTH_LONG).show();
    Log.e("errorVolely", error.toString());}
```

Il·lustració 48 : Codi Volley III. FONT: Pròpia

En aquest fragment de codi, permet comprovar quin error s'ha produït en la comunicació, utilitzant el Log (Logcat, és una eina de *debuggeig* que ofereix Android Studio) i informar a l'usuari per mitjà d'un *Toast Message*.

Les dades que s'envien mitjançant les sol·licituds POST necessiten un mètode més, el *getParams*.

```
@Override
protected Map<String, String> getParams() {
    Map<String, String> postparametres = new HashMap<>();
    postparametres.put("nickname", nickname);
    postparametres.put("pass", password);
    return postparametres;}
```

Il·lustració 49 : Codi Volley IV. FONT: Pròpia

Aquest mètode permet emmagatzemar informació en format clau-valor i retorna l'objecte a Volley. La clau `"nickname"` i `"pass"` s'utilitzarà per capturar els valors mitjançant la funció $_POST["clau"] en els scripts de PHP que es descriuen en l'apartat 5.2.1. *Scripts* amb php.

Per poder accedir al *script* especificat en l'URL, serà necessària **una autentificació.** Per això, s'ha d'incloure el mètode ***getHeaders().***

```
@Override
public Map<String, String> getHeaders()  {
    Map<String, String> headers = new HashMap<>();
    String user000webhost=getString(R.string.user_000webhost);
    String pass000webhost=getString(R.string.pass_000webhost);
    String credentials=user000webhost+":"+pass000webhost;
    String auth = "Basic " + Base64.encodeToString(credentials.getBy
tes(), Base64.NO_WRAP);
    //headers.put("Content-Type", "application/json");
    headers.put("Authorization", auth);
    return headers;
}
```

Il·lustració 50 : Codi Volley V. FONT: Pròpia

Aquest mètode és capaç d'autentificar la connexió amb una contrasenya i usuari. Aquests dos han d'estar codificats en **Base64**. La taula d'aquesta codificació es pot trobar aquí.

Sol·licitud GET:

Per implementar una sol·licitud d'aquest tipus es necessari l'ús de *JSONOBJECTREQUEST*:

```
String base_dades_remota=getString(R.string. ip_000webhost);
String URL = base_dades_remota.getIp()+"/ConsultaUsuari.php?nick-
name="+nickname;
URL=URL.replace(" ","%20");
jsonObjectRequest=new JsonObjectRequest(Request.Method.GET,URL,...){}
```

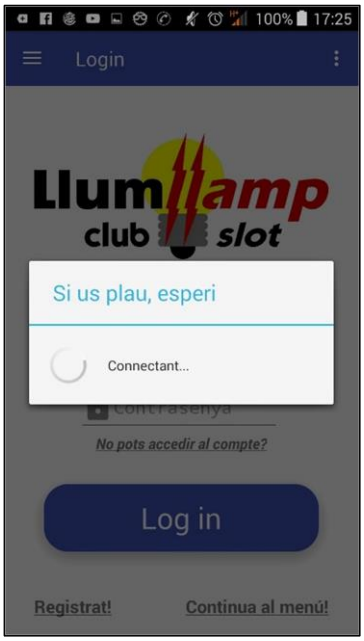
Il·lustració 51 : Codi Volley VI. FONT: Pròpia

En la petició get, les dades es transmeten com a part de l'URL, per tant, el mètode *getParams* ja no serà necessari. Seran imprescindibles els mètodes de**:** ***onResponse, onErrorResponse i getHeaders*** el quals actuaran de la mateixa forma que en la sol·licitud POST a excepció, del *onResponse*, que ara espera rebre un objecte del tipus JSON. També s'ha d'especificar al mètode getHeaders que la petició és de tipus JSON descomentant la sentència de l'exemple anterior: `//headers.put("Content-Type", "application/json");`

5.1.7 Ús de tasques asíncrones

Les tasques asíncrones són una eina molt important de les aplicacions. Tots els components i tasques s'introdueixen en el fil principal o *thread* de l'UI (*main thread*). El problema sorgeix quan es volen executar tasques que duren varis segons en finalitzar, aquestes poden fer que l'aplicació acabi bloquejant-se. Les tasques asíncrones permeten paral·lelisme entre tasques dins de l'aplicació.

Al projecte, es fa ús de les *AsyncTask* per iniciar una petició i rebre la resposta provinent del servidor. Un cop es rep la resposta es modifica la IU amb els canvis corresponents.



Il·lustració 52 : Progress Dialog AsyncTask I. FONT: Pròpia

A continuació es mostrarà un exemple d'implementació:

```
private class TaskConsultaUsuariLogin extends AsyncTask<String, Void,
String> {
    @Override
    protected void onPreExecute(){
        progressDialog = new ProgressDialog(getActivity());
        progressDialog.setTitle("Si us plau, esperi");
        progressDialog.setMessage("Connectant...");
        progressDialog.setCancelable(false);
        progressDialog.show();
    }
    @Override
    protected String doInBackground(String... strings) {
        //funció a executar
        return "TasKConsultaUsuariLoginCompleted";
    }
    @Override
    protected void onPostExecute(String result){
        super.onPostExecute(result);
    }
}
```

Il·lustració 53 : Codi AsyncTask I. FONT: Pròpia

Per iniciar la tasca asíncrona s'ha d'utilitzar la funció *execute*(). Aquesta permet el pas de paràmetres i es capaç de retornar un valor. D'aquests mètodes cal destacar-ne el ***onPreExecute()***, aquest s'executa abans que el mètode doInBackground i permet, com en l'exemple, obrir un *progress dialog* el qual informarà l'usuari que ha d'esperar-se durant la resolució de la petició.

La funció ***doInBackground()*** serà el mètode on s'executaran les tasques o funció necessàries del fil. S'executa en paral·lel amb el *thread* de la IU.

onPostExecute() Aquest mètode s'invoca automàticament després del doIn-Background(), si la tasca es cancel·lada no s'invocarà.

5.1.8 *Shared preferences* (Preferències)

Les *Shared Preferences* permeten emmagatzemar dades en els dispositius en format de clau valor, és a dir, podem definir una clau que es digui "user_pass" i associar-hi la contrasenya que ha introduït l'usuari.

L'objectiu d'utilitzar les preferències en el projecte és **facilitar l'inici de sessió**. Per a no forçar a introduir el *nickname* i la contrasenya cada cop que s'inicia l'aplicació, aquests paràmetres s'emmagatzemen en l'arxiu *preferences.xml* de l'app. L'aplicació consulta aquests valors e inicia la sessió automàticament enviant una petició al servidor.

```
SharedPreferences.Editor editor = getSharedPreferences(MY_PREFS_NAME,
MODE_PRIVATE).edit();
editor.putString("user_name",user_name);
editor.putString("user_pass",pass);
editor.apply();
```

Il·lustració 54 : Exemple emmagatzemar Shared Preferences. FONT: Pròpia

5.1.9 Generar document PDF amb iTEXTPDF

En aquest apartat s'explicarà com s'ha generat el document PDF de la fitxa tècnica del vehicle. Per fer-ho possible s'utilitza la classe ***templatepdf***, la qual, fa ús de funcions de la llibreria iTEXTPDF. A continuació s'exposarà el codi il·lustratiu encarregat de fer-ho:

```
TemplatePDF templatePDF=new TemplatePDF(getActivity());
String[]header={"Component","Material"};
String shortText="Especificacions tècniques per al següent cotxe: ";
ArrayList<String[]>rowsCotxe=new ArrayList<>();
String date = new SimpleDateFormat("dd-MM-yyyy", Locale.getDefault()).format(new
Date());
templatePDF.openDocument(Cotxeactual.getNomConfig());
templatePDF.addMetaData(Cotxeactual.getNomConfig(),"Fitxa Tècnica Cotxe
Slot","autor_llumllampapp");
templatePDF.addTitles(Cotxeactual.getNomConfig(),Cotxeactual.getEscala()+" :
"+Cotxeactual.getCategoria(),date);
templatePDF.addImage(bmp,1);
templatePDF.addParagrapgh(shortText);
templatePDF.createTableMaterial(header,rowsCotxe);
templatePDF.addParagrapgh("Pdf generat amb l'aplicació de LlumLlamp Club Slot");
Bitmap bitmap = BitmapFactory.decodeResource(this.getResources(),
R.drawable.logo_llumllamp);
ByteArrayOutputStream stream = new ByteArrayOutputStream();
bitmap.compress(Bitmap.CompressFormat.JPEG, 100, stream);
templatePDF.addImage(bitmap,5);
templatePDF.closeDocument();
templatePDF.viewPDF(getActivity());
```

Il·lustració 55 : Generació de PDF classe TemplatePDF. FONT: Pròpia

Aquest codi es troba dins d'una funció que actua un cop es fa *click* sobre el botó de generar PDF. Primerament és imprescindible fer una instància de la classe *TemplatePDF* que rebrà com a paràmetre el context de l'activitat. Posteriorment s'inicialitza un *arraylist* on **s'introduirà els valors dels components que formin la taula de material.** Els valors de cadascun dels **paràmetres introduïts** es trobaran dins d'una instància de la classe **Cotxe.**

Per afegir un títol al document s'utilitza el mètode *OpenDocument()*, el qual, se li passa el nom de la configuració. L'ús del mètode *addMetaData(títol, tema, autor)* s'utilitza per afegir **l'autoria**. Aquests són visibles quan es consulta les propietats del document.

A partir d'aquí, s'afegeix el contingut: títols, imatge, fragment de text, la taula de material, un altre fragment de text i el logotip de LlumLLamp. Automàticament, mitjançant el mètode ***ViewPDF()*** s'obre l'arxiu generat utilitzant l'aplicació predeterminada per aquest tipus de document.

5.2 Elements del servidor

Dins d'aquest apartat es tractaran els elements més rellevants de la implementació del disseny sobre el servidor de **000webhost**.

S'explicarà com actuen els *scripts* amb PHP, com s'envien els correus amb PHPMailer i com un administrador pot gestionar la base de dades amb phpMyAdmin.

5.2.1 Scripts amb PHP

Dels scripts amb PHP es poden distingir dos tipus principalment: Els que estan encarregats de rebre peticions des de l'aplicació i els que reben un codi provinent dels enllaços d'activació, canvi de correu o recuperar contrasenya. Els primers es troben (dins del servidor públic) en el directori bd_Slot_club mentre que els segons en bd_Slot_club_email.

Tots els paràmetres provinents tant de sol·licituds GET/POST s'obtenen i es comproven amb la funció ***isset().***

```
if(isset($_POST["nickname"])&&isset($_POST["pass"])){
```

Il·lustració 56 : Codi funció Isset. FONT: Pròpia

Primerament, s'obre la connexió a la base de per poder gestionar-la, utilitzant la funció ***Connect_to_mysql()****,* importada des del mòdul ***database_mysql.sql*****,** en aquest mòdul, es disposa d'una funció que retorna la connexió a la base de dades:

```php
$conexion=Connect_to_mysql();
$escala=mysqli_real_escape_string($conexion,$_POST['escala']);
$categoria=mysqli_real_escape_string($conexion,$_POST['categoria']);
$marcacotxe=mysqli_real_escape_string($conexion,$_POST['marcacotxe']);
$userPropietari=mysqli_real_escape_string($conexion,$_POST['userPropietari']);
```

Il·lustració 57 : Codi escapament de variables PHP. FONT: Pròpia

Posteriorment, s'emmagatzemen els paràmetres en variables globals escapant el seu valor amb les funcions ***mysqli_real_escape_string()*** *i* ***intval*****().** Aquestes tenen l'objectiu d'evitar que es produeixin injeccions SQL.
Per exemple, un usuari podria crear-se un nom amb la següent forma: *1=1;DROP ALL DATABASES;* → d'aquesta manera s'eliminaria la informació de totes les taules.

A partir d'aquest punt ja es poden executar consultes, inserts o updates amb els paràmetres totalment depurats:

```php
$consulta="select * from rally where rally_edicio='{$edicio}' and rally_nom='{$rally}'";
$resultat=mysqli_query($conexion,$consulta);
```

Il·lustració 58 : Codi sentència MySQL PHP. FONT: Pròpia

Per obtenir les columnes i files retornades per la base de dades es disposen de funcions com ***mysqli_fetch_array*** *o* ***mysqli_affected_rows*** *.* La primera permet obtenir les columnes retornades i la segona retorna el nombre de columnes modificades en l'última sentència SQL (*insert o update*).

Per poder retornar la informació en format JSON, es tan simple com executar la funció de *json_encode*, la variable *$json* és un *array*:

```php
if($registre=mysqli_fetch_array($resultatselect)){
    $json['cotxe'][]=$registre;
    echo json_encode($json);
```

Il·lustració 59 : Codi json_encode. FONT: Pròpia

5.2.2 Enviament automàtic de correu amb PHPMailer

L'encarregat d'enviar els missatges de correu als usuaris de l'aplicació és el **servidor**. Disposa d'un mòdul anomenat ***Envia_Correu.php***. Aquest utilitza la llibreria de phpMailer per crear una instància d'objecte ***PHPMailer().*** Aquest mòdul disposa de funcions que seran cridades des de diversos *scripts* quan sigui necessari.

Envia_correu.php disposa de 4 funcions amb missatges diferents per enviar a l'usuari. Els mòduls que criden aquestes funcions són: ***Registre_usuari.php, EnviaCanviCorreu.php, Registre_Inscripcio.php, EnviaInfoAccount.php.***

```php
$email_user = //email a utilitzar
$email_password = //contrasenya del correu a utilitzar
$the_subject = "Autentificar el compte a LlumLlamp Club Slot App"; //títol del missatge
$address_to = $email; //email a enviar el correu
$from_name = "LlumLlamp Club Slot"; //nom de l'emissor del correu
$phpmailer = new PHPMailer();//crea nou objecte PHPmailer
// ---------- dades GMAIL ACCOUNT ------------------------------
$phpmailer->Username = $email_user;
$phpmailer->Password = $email_password;
//------------------------------------------------------------------------
$phpmailer->SMTPSecure = 'ssl'; //seguretat Secure Sockets Layer
$phpmailer->Host = "smtp.gmail.com"; // utilitza servidor de gmail
$phpmailer->Port = 465;
$phpmailer->Helo="smtp.gmail.com";
$phpmailer->IsSMTP(); // utilitza SMTP
$phpmailer->SMTPAuth = true; //autenticació SMTP
$phpmailer->setFrom($phpmailer->Username,$from_name); //qui envia el missatge?
$phpmailer->AddAddress($address_to); // destinatari del missatge
$phpmailer->Subject = $the_subject; //assigna títol del missatge.
$phpmailer->CharSet = 'UTF-8'; //codificació del missatge
//Contingut del missatge
$message='<html><body>';
//host image Llumllamp Logo Notificacions
$message.='<img src="https://i.imgur.com/ArQCKew.png" alt="Logo Notificacions LlumLLamp" />';
$message.='<p><h1>Benvingut/da <strong>'.$nickname.'!</strong></h1></p>';
$message.='<p>Per verificar que el compte de correu <strong>'.$email.'</strong> que has utilitzat és realment teu, has de fer
click sobre aquest enllaç o copiar la URL al navegador.</p>';
$message.='<p><a href="'.$base_url.'activation.php?code='.$activation.'">'.$base_url.'activation.php?code='.$activation.'</a></p>';
$message.='<p>Esperem que disfrutis fent servir aquesta aplicació.</p>';
$message.='<p>Atentament, Llum Llamp Club Slot.</p>';
$message."<p>Data i hora: ".date("d-m-Y H:i:s")."</p>";
$message.='</body></html>';
$phpmailer->Body=$message;

$phpmailer->IsHTML(true);//permet contingut HTML
$exit = $phpmailer->Send(); //envia el missatge
if($exit){  $result = true;
        return $result;}
else{   $result = false;
    return $result;}
```

Il·lustració 60 : Enviar correu amb PHPMailer. FONT: Pròpia

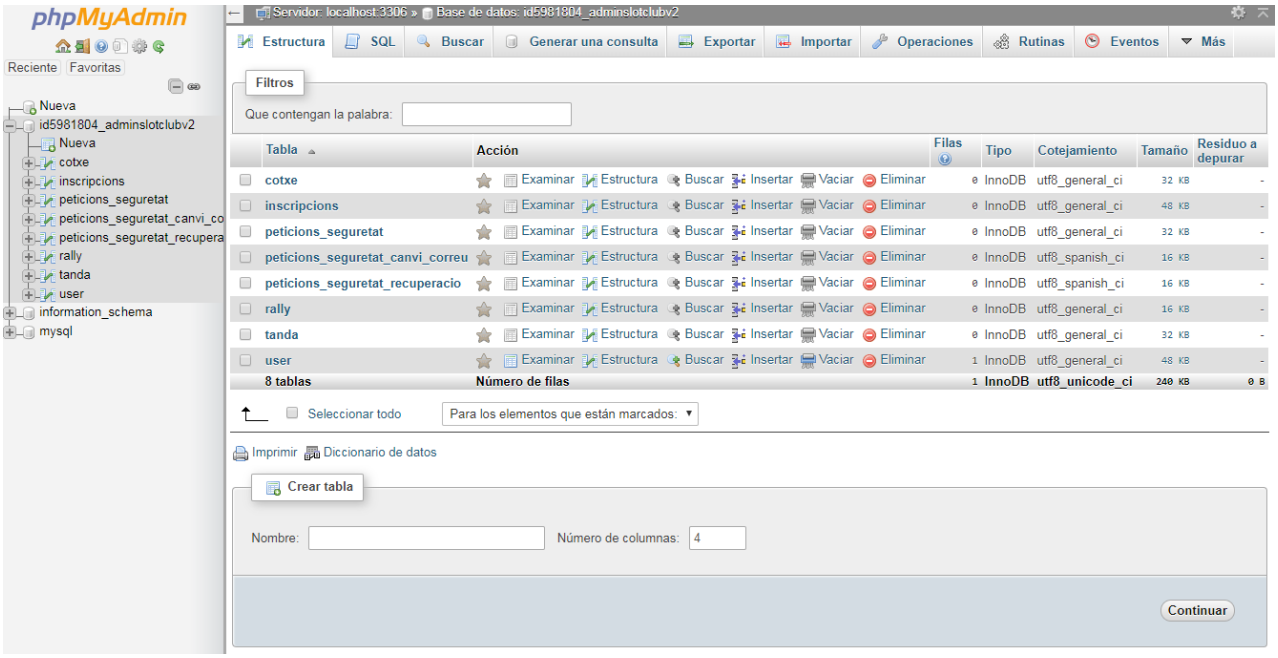
5.2.3 Gestió de la base de dades remota amb phpMyAdmin

El servei web de 000webhost.com disposa de gestió de la base de dades per part dels administradors. Per poder accedir-hi és necessari un usuari i contrasenya per realitzar la connexió de MySQL. El format del nom de la base de dades i de l'usuari sempre serà d'aquest estil:

Nombre de Base de Datos	Usuario de Base de Datos	Host de Base de datos	
id5981804_adminslotclubv2	id5981804_adminllumllampv2	localhost	Gestionar ▾

Il·lustració 61 : Panell de control I. FONT: Pròpia

000webhost ofereix el panell de control phpMyAdmin 4.7.7. Des d'aquest panell es pot realitzar qualsevol acció sobre la base de dades. Per exemple, es pot eliminar un usuari o una tanda entre altres. La interfície té el següent aspecte:



Il·lustració 62 : Panell de control II. FONT: Pròpia

phpMyAmin permet veure quin motor de base de dades s'està utilitzant, la codificació dels paràmetres (UTF-8) i el pes de la taula. Com **a valor aproximat**, mostra el nombre de files que sumen totes les taules.

6. Proves del sistema i avaluació

A continuació es mostraran les proves que s'han realitzat al llarg del projecte. S'ha anat testejant a mesura que s'anava afegint funcionalitats per tal de comprovar la correcta implementació. A continuació s'explicaran els casos problemàtics que pot realitzar **l'usuari**, com s'han tractat i quina es la sortida esperada.

També, en la segona part, s'inclourà un petit resum de quines han sigut les versions on el sistema ha sigut *testejat* per validar el seu funcionament i **l'opinió dels usuaris**, en aquest cas, la directiva de Llum Llamp Club Slot al veure l'aplicació.

6.1 Casos de prova

Els casos de prova són situacions que esperen una **entrada,** l'*input* es tractat pel sistema (verificat i depurat). Un cop ha passat tot el procés, s'informa **l'usuari quin ha estat el resultat o en cas contrari, quin ha estat l'error.**

S'ha recollit els principals casos de prova a l'aplicació en aquest llistat:

- Log in
- Registre usuari
- Editar perfil
- Editar Contrasenya
- Canvi de correu
- Recuperació del compte
- Consulta preinscrits
- Configuració comandament
- Inscripció a la cursa
- Generar fitxa tècnica del vehicle.

En cada un dels casos s'especifica quin és el propòsit, els prerequisits, les dades d'entrada, el conjunt de passos per resoldre el cas i el resultat esperat.

A continuació es mostrarà en format de taula, cadascun d'ells.

CAS DE PROVA	LOG IN
Propòsit	Iniciar sessió d'usuari.
Prerequisits	Ha d'existir l'usuari.
Dades d'entrada	• Usuari • Contrasenya
Passos per resoldre el cas	1. Comprovació dels paràmetres d'entrada. 2. Es verifica l'existència de l'usuari introduït 3. Es comprova que l'usuari introduït i la contrasenya són les correctes.
Resultat esperat	Si es superen tots els passos, s'inicia sessió. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 7 : Cas de prova Login. FONT: Pròpia

CAS DE PROVA	REGISTRE USUARI
Propòsit	Registrar un usuari nou.
Prerequisits	-
Dades d'entrada	• Usuari • Contrasenya • Repetir Contrasenya • Correu
Passos per resoldre el cas	1. Comprovació dels paràmetres d'entrada. 2. Es verifica si hi ha un usuari amb aquest nom i correu. 3. S'envia un correu d'activació.
Resultat esperat	Si es superen els passos 1 i 2, es registre l'usuari. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 8: Cas de prova Registre Usuari. FONT: Pròpia

CAS DE PROVA	EDITAR PERFIL
Propòsit	Actualitzar la informació de l'usuari.
Prerequisits	Ha d'existir l'usuari.
Dades d'entrada	• Usuari • Nom • Cognom • Escuderia • Telèfon
Passos per resoldre el cas	1. Comprovar tots els paràmetres d'entrada. 2. Es comprova que l'usuari a actualitzar existeix. 3. Es verifica que no existeix un usuari amb cap nom igual.

Resultat esperat	Si es superen tots els passos, s'actualitza l'usuari. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 9 : Cas de prova editar perfil. FONT: Pròpia

CAS DE PROVA	EDITAR CONTRASENYA
Propòsit	Actualitzar la contrasenya de l'usuari.
Prerequisits	Ha d'existir l'usuari.
Dades d'entrada	• Contrasenya • Repetir Contrasenya
Passos per resoldre el cas	1. Comprovar tots els paràmetres d'entrada. 2. Es verifica que existeixi l'usuari.
Resultat esperat	Si es superen tots els passos, s'actualitza la contrasenya. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 10 : Cas de prova editar contrasenya. FONT: Pròpia

CAS DE PROVA	CANVI DE CORREU
Propòsit	Canviar la direcció de correu.
Prerequisits	Ha d'existir l'usuari.
Dades d'entrada	• Correu • Repetir Correu
Passos per resoldre el cas	1. Comprovar tots els paràmetres d'entrada. 2. Es verifica que existeixi l'usuari. 3. Es comprova que cap usuari utilitza el nou correu que es vol assignar. 4. S'envia un e-mail al correu amb un codi de confirmació del canvi.
Resultat esperat	Si es superen els passos (1, 2, 3) s'envia el codi de confirmació al correu. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 11 : Cas de prova canvi de correu. FONT: Pròpia

CAS DE PROVA	RECUPERACIÓ DEL COMPTE
Propòsit	Recuperar les credencials del compte.
Prerequisits	Ha d'existir l'usuari.
Dades d'entrada	• Correu
Passos per resoldre el cas	1. Comprovar el paràmetre d'entrada. 2. Es verifica que existeixi un usuari amb aquest correu. 3. S'envia un e-mail al correu per iniciar un procés de restauració

Resultat esperat	Si es supera els passos 1 i 2 s'envia el correu amb un codi de recuperació. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 12 : Cas de prova recuperació del compte. FONT: Pròpia

CAS DE PROVA	CONSULTA PREINSCRITS
Propòsit	Obtenir el llistat de preinscrits per una cursa.
Prerequisits	L'usuari ha d'estar *loggejat*.
Dades d'entrada	• Escala • *Rally* • Edició
Passos per resoldre el cas	1. Comprovar tots els paràmetres d'entrada. 2. Es comprova que el *rally* està disponible. 3. S'obté els inscrits a la cursa.
Resultat esperat	Si es supera els passos 1 i 2 es mostra una llista de preinscrits. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 13 : Cas de prova consultar preinscrits. FONT: Pròpia

CAS DE PROVA	CONFIGURACIÓ COMANDAMENT
Propòsit	Generar una nova configuració/actualitzar informació.
Prerequisits	L'usuari ha d'estar *loggejat* o mode desconnectat.
Dades d'entrada	• Nom configuració • Acceleració • Freno • *Anti-spin* • Voltatge
Passos per resoldre el cas	1. Comprovació dels paràmetres. 2. Verificar si existeix una configuració amb aquell nom. 3. Generar / actualitzar la informació.
Resultat esperat	Si es supera els passos 1 i 2 s'actualitza o es genera una configuració. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 14 : Cas de prova configuració comandament. FONT: Pròpia

CAS DE PROVA	INSCRIPCIÓ A LA CURSA
Propòsit	Generar una inscripció a la cursa.
Prerequisits	L'usuari ha d'estar *loggejat*.
Dades d'entrada	• Escala • *Rally* • Tanda • Categoria

	• Marca del vehicle
Passos per resoldre el cas	1. Comprovació de tots els paràmetres d'entrada 2. Verificar que l'usuari té el compte activat. 3. Comprovar si l'usuari ja té un vehicle enregistrat a la base de dades, en cas contrari, s'assigna un nou dorsal al vehicle. 4. Comprovar que el *rally* està disponible.´ 5. Comprovar si el *rally* té la tanda disponible. 6. S'envia un correu informatiu on queda el resguard de la inscripció.
Resultat esperat	Si es supera els passos 1, 2, 3 , 4, 5 es mostra una llista de preinscrits. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 15 : Cas de prova inscripció. FONT: Pròpia

CAS DE PROVA	GENERAR FITXA TÈCNICA VEHICLE
Propòsit	Generar una fitxa tècnica del vehicle
Prerequisits	L'usuari ha d'estar *loggejat*.
Dades d'entrada	• Nom configuració • Escala • Categoria • Fotografia • Paràmetres del vehicle
Passos per resoldre el cas	1. Comprovar almenys s'ha introduït el nom de la configuració, escala, categoria. 2. Comprovar que no existeix un altre document amb aquell mateix nom. 3. Generar un document PDF amb tots els atributs assignats.
Resultat esperat	Si es supera els passos 1, 2, es genera un document PDF. En cas contrari, informa a l'usuari quin pas no ha superat.

Taula 16 : Cas de prova generar fitxa tècnica. FONT: Pròpia

Per últim, s'ha *testejat* casos de manera concurrent des de dos dispositius mòbils alhora. El resultat ha sigut favorable sempre, ja que s'utilitza el motor **innoDB** per emmagatzemar la informació. Aquest **garanteix la concurrència** i les transaccions de tipus **ACID.** A més a més, els bloquejos en casos de concurrència són molt baixos perquè permet una **granularitat de bloqueig** a **nivell de fila** d'aquesta manera, no queda bloquejada una taula sencera fins que acabi la transacció. En cas d'error en una transacció s'inicia un *rollback* de la base de dades a l'anterior versió.

6.2 Test de l'aplicació

Per poder comprovar el correcte funcionament de l'aplicació sobre diferents versions d'Android, s'ha *testejat* en els següents dispositius físics:

TELÈFON MÒBIL	VERSIÓ ANDROID
Aquaris e5 hd	**4.4.2**
Samsung Galaxy S7	**6.0.1**
Samsung Galaxy J3	**5.1.1**
Samsung Galaxy S3	**4.2.2**

Taula 17 : Test sobre dispositius físics. FONT: Pròpia

També, s'ha aconseguit testejar el funcionament del sistema sobre els dispositius Android emulats que es mostraran a continuació.

EMULADOR	VERSIÓ ANDROID
Nexus 5	**8.0**
Pixel 2	**8.0**
Nexus 5X	**4.4**

Taula 18 : Test sobre emuladors. FONT: Pròpia

En tots els dispositius el funcionament del projecte és correcte, però, es mostra una **petita latència a l'obrir el fragment de gestió_cotxe** en els **dispositius amb una versió inferior a 5**.

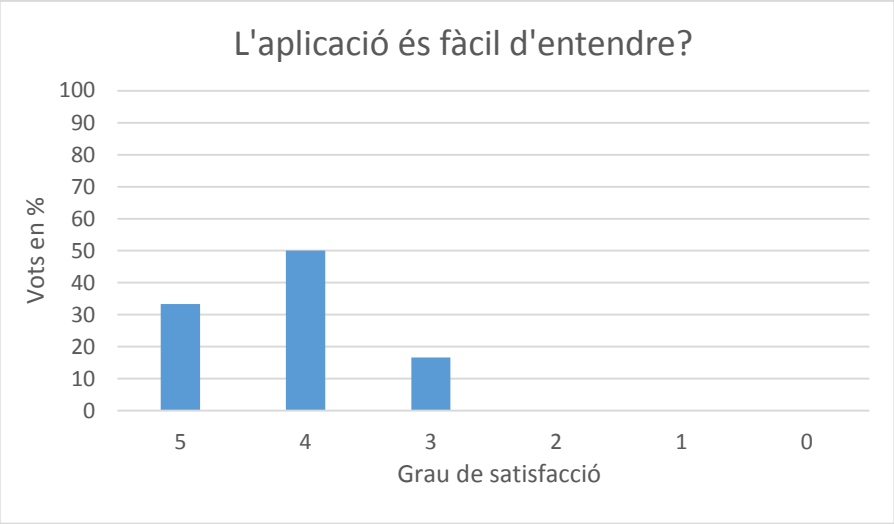
6.3 Opinió i satisfacció dels usuaris

Un factor imprescindible en analitzar, és conèixer si l'aplicació desenvolupada compleix amb les expectatives i requisits que necessitava el client. També conèixer el grau de satisfacció de l'usuari.

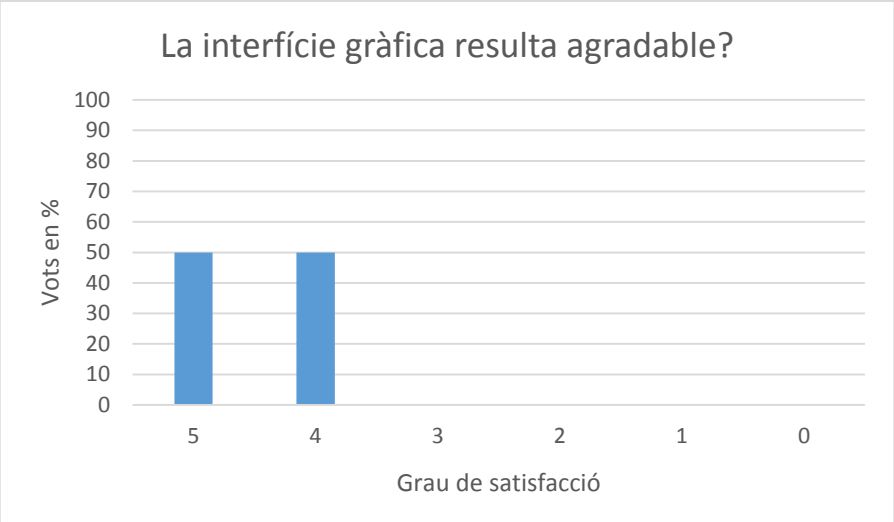
S'ha realitzat una enquesta a la directiva i cercle més proper de Llum Llamp (6 persones) per tal de conèixer la seva opinió i valoració del projecte. En aquesta enquesta s'ha plantejat les següents preguntes:

- L'aplicació és fàcil d'entendre?
- La interfície gràfica resulta agradable?
- Creus que l'aplicació compleix amb els requeriments?
- Què és el que més t'agrada de l'aplicació?
- Creus que pot ser atractiva per un públic no únicament de Llum Llamp?

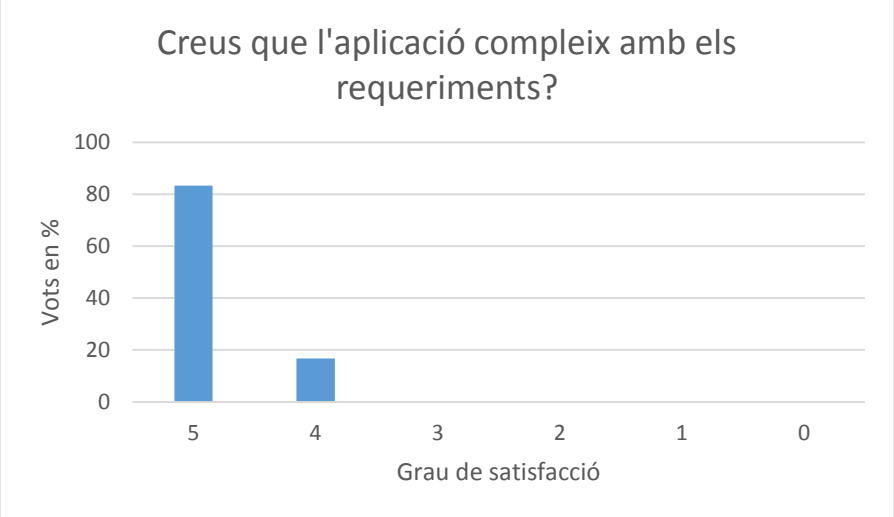
A continuació es mostren els resultats obtinguts, **el grau de satisfacció es puntua de 0 a 5, on 5 és molt d'acord i 0 és molt en desacord**:



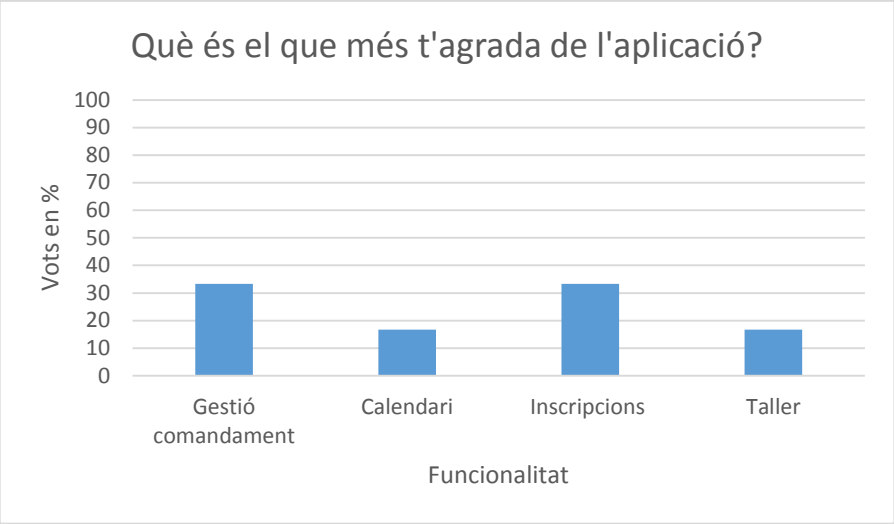
Il·lustració 63 : pregunta 1. FONT: Pròpia



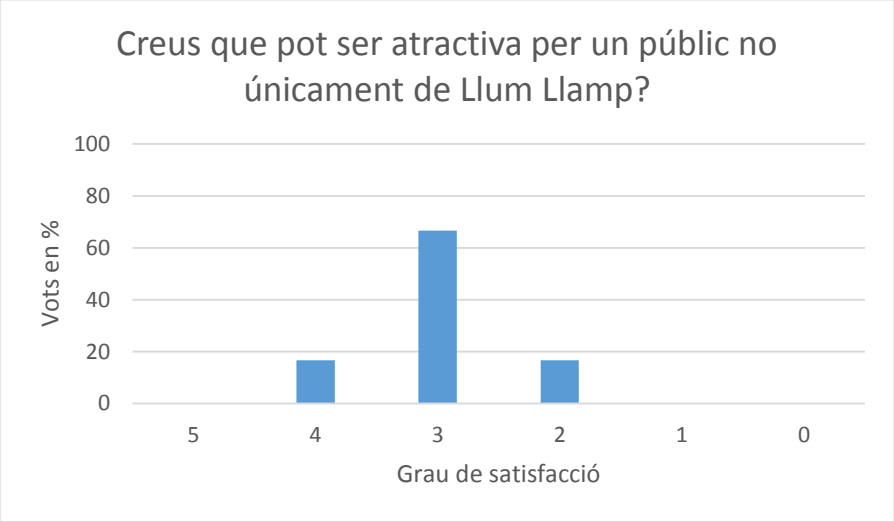
Il·lustració 64 : pregunta 2. FONT: Pròpia



Il·lustració 65 : pregunta 3. FONT: Pròpia



Il·lustració 66 : pregunta 4. FONT: Pròpia



Il·lustració 67 : pregunta 5. FONT: Pròpia

Després de veure la resposta obtinguda es pot extreure les següents conclusions:

- Els usuaris estan satisfets amb la forma en què estan dissenyades les seves funcionalitats i la facilitat de fer-les funcionar.
- La majoria dels usuaris estan molt satisfets respecte el disseny gràfic de l'aplicació.
- Els requeriments proposats s'han assolit favorablement.
- Respecte a quina funcionalitat agrada més, hi ha un equilibri interessant entre les que es van plantejar com a **principals de l'aplicació**. No ha rebut cap vot: *streaming* de la cursa, consultar resultats.
- Conseqüentment, el fet de tenir una aplicació tan ajustada per un club determinat, limita l'abast de població que l'arribaria a utilitzar.

7. Pressupost de l'aplicació

A continuació es tractarà el pressupost de l'aplicació en un possible cas de comercialització. Per desenvolupar un projecte com aquest, es necessiten mitjans ***hardware i software*** suficients per tenir un bon entorn de treball productiu. A més a més**, el principal cost del projecte estarà centrat en els recursos humans.**

Al llarg del treball, com a autor, he assumit diferents funcions: **analista** de les principals branques de desenvolupament, **programador** de l'aplicació i per últim, encarregat de **documentar** tota la informació del projecte.

Cadascuna d'aquestes funcions té una remuneració diferent. S'utilitzarà els següents preus de mà d'obra per realitzar un càlcul estimat del pressupost.

1. Un **analista**, per realitzar les funcions de recopilació d'informació, d'anàlisis i disseny, cobraria uns **17€/hora**.
2. Un **programador**, per realitzar la feina d'implementació, proves, retocs i *testejos*. **15€/hora**
3. Una persona encarregada de recopilar totes les activitats relacionades al llarg del treball i **redactar-ho** en una memòria de qualitat. **11€ /hora.**

El projecte es va iniciar el **29 de gener de 2018** i ha finalitzat aproximadament a principis del mes de juliol, s'estimarà el dia 1. Durant tot aquest període, s'ha realitzat un treball de **38,5 hores/setmanals**.

Estimació d'hores de treball en els diferents camps:

TASQUES	HORES DE TREBALL	DIES DE TREBALL (5,5h/dia)
Recopilació d'informació	99 h	18
Anàlisis	126,5h	23
Disseny	104,5h	19
Implementació	423,5h	77
Proves, *testejos* i retocs	60,5h	11
Documentació	60,5h	11
TOTAL	**874,5h**	**153 dies**

Taula 19 : Pressupost mà d'obra. FONT: Pròpia

Pressupost mà d'obra : 330h d'anàlisis * 17€/hora + 484h programació * 15€/hora + 60,5h * 11€/hora = 5610€ + 7260€+ 665,5€ = **13535,5 €.**

El pressupost relacionat amb el *hardware* és el següent. Cal destacar, que per a cada element, s'especificarà quin percentatge proporcional d'ús s'ha utilitzat dins del projecte.

HARDWARE	COST TOTAL	% ÚS PROJECTE	PREU REAL
Ordinador	879€	40%	351,6€
Dispositiu Android	200€	20%	40€
Material oficina	20€	100%	20€
Connexió a Internet	200€ (5 mesos)	20%	40
TOTAL			**451,6€**

Taula 20 : Pressupost del hardware. FONT: Pròpia

Després de calcular el percentatge d'ús per a cada element, **el pressupost del hardware és de 451,6€.**

Els elements que conformen les despeses relacionades amb el *Software*, es poden veure a continuació:

SOFTWARE	COST TOTAL	% ÚS PROJECTE	PREU REAL
Llicència Windows 7	20,95€	15%	3,15€
Llicència Ubuntu 16.04	Gratuït	100%	Gratuït
SQLite3 i MySQL	Gratuït	70%	Gratuït
Llicència SQLyog *Free*	Gratuït	100%	Gratuït
Android Studio i SDK	Gratuït	100%	Gratuït
XAMPP Server	Gratuït	100%	Gratuït
000webhost	Gratuït	100%	Gratuït
Twitch i Obs Studio	Gratuït	100%	Gratuït
TOTAL			**3,15€**

Taula 21 : Pressupost del software. FONT: Pròpia

El pressupost relacionat amb el ***software*** és de **3,15€.** Un cop analitzat aquests camps es pot concloure que el pressupost total per a desenvolupar l'aplicació ha sigut el següent:

	COST
Mà d'obra	13535,5 €
Hardware	451,6€
Software	3,15€
TOTAL	**13990,25€**

Taula 22 : Pressupost total de l'aplicació. FONT: Pròpia

8. Millores de futur

El producte actual és una primera versió de l'aplicació. La plataforma d'Android ofereix un camp tant ampli de possibilitats que permet millorar l'aplicació amb noves idees i eines.

Un dels principals punts a resoldre és **la traducció** de la interfície gràfica i dels missatges que proporciona el sistema per notificar a l'usuari. L'aplicació va dirigida principalment a un públic català, per això, està íntegrament en aquest idioma, però, caldria expandir-la al màxim d'idiomes possibles. Per exemple, al castellà i l'anglès.

Un aspecte a tractar és la millora de **la interfície gràfica** amb l'ajuda d'un especialista es podria aconseguir un acabat visual més professional. Es podria dissenyar logotips exclusius i adaptats per l'aplicació. D'aquesta manera es podria adaptar els *layouts* a les diferents resolucions de pantalles actuals sense perdre qualitat.

Un element important seria **generar notificacions quan s'apropés la data de la pròxima cursa** per informar a l'usuari, que aquell cap de setmana hi ha una competició.

La **migració del servei web de 000webhost** al que actualment està utilitzant Llum Llamp Club Slot (lasequia.cat), el qual, no té restriccions d'ús. Això, pot suposar una millora important en el rendiment de temps d'espera de la comunicació entre el client i el servidor.

Per millorar la seguretat en les comunicacions s'haurà d'afegir un **certificat SSL** a Volley per establir comunicacions cent per cent segures vers atacs.

Respecte a les millores relacionades amb **funcionalitats de l'aplicació**, es podria destacar varis punts:

- Incloure a l'entrenament una funcionalitat que permeti emmagatzemar el temps per volta i el temps global que s'ha obtingut al acabar un tram.
- Incloure una funcionalitat tal que es pugues anotar quin temps vas realitzar la passada anterior i es calcules automàticament, quan temps has millorat o empitjorat al llarg dels trams que s'introdueixen.
- Permetre emmagatzemar la foto del comandament al qual esta assignada una configuració.
- Desenvolupar un xat exclusiu pels socis on es pugessin parlar i tractar temes del club.

Un apartat interesant, seria aconseguir publicar a la **Play Store** l'aplicació desenvolupada. D'aquesta manera s'aconseguiria que tots els socis o persones interessades tinguessin accés a ella.

9. Conclusions personals

El projecte realitzat respon a les expectatives i requeriments que es van plantejar en el punt de partida de l'anàlisi. Finalment s'ha aconseguit crear una aplicació mòbil per Android ajustada al club d'Slot Llum Llamp.

Respecte al desenvolupament d'aplicacions per mòbil va suposar un **inici molt dur** perquè no havia programat prèviament amb Java i **no coneixia pràcticament res del món d'Android**. Tot i això, gràcies a les assignatures que vaig cursar al llarg del grau, m'han permès establir una base per obrir-me a nous llenguatges de programació. Cal destacar, que existeix una gran comunitat de desenvolupadors d'Android que ofereixen una documentació excel·lent per resoldre dubtes i oferir-te punts de vista molt diferents sobre com fer les coses.

Durant el projecte, s'han anat modificant i retocant parts que pensava que ja estaven acabades implementant noves tècniques que anava descobrint. També, per tal d'evitar problemes d'incompatibilitat entre dispositius. Per tant, **el fet de refer alguns elements per millorar les prestacions** ha sigut un aspecte a tenir molt en compte en el projecte.

L'entorn de desenvolupament **d'Android Studio ha sigut una eina imprescindible** per a un programador nou en aquesta plataforma com jo, que vol obrir-se pas dins d'aquest món. Aquest entorn és capaç de suggerir-te idees, estructures de codi... Que ajuden moltíssim si encara no coneixes en profunditat la plataforma.

Mirant cap a la banda del servei web desenvolupat a **000webhost, el qual disposa de panells de control i eines pròpies**, facilita moltíssim l'administració de les bases de dades i gestió dels arxius.

Després de gairebé cinc mesos, finalment he aconseguit concloure el treball final de grau. **He après com programar en Java i PHP**, llenguatges totalment desconeguts per a mi, a més a més, d'aprendre com funciona el desenvolupament d'aplicacions en sistemes Android.

Estic totalment satisfet amb el resultat aconseguit i per tots els coneixements que he après durant aquest projecte. Actualment l'aplicació està acabada en aquesta primera versió i a punt de ser publicada a la Play Store.

10. Referències

A continuació es llistarà la *webgrafia* que s'ha utilitzat per dur a terme el projecte. Aquestes referències es dividiran principalment en dos grups. En el primer, consultes web, on es troben anotades les principals fonts d'informació. En el segon, s'anotarà la font de les imatges que no són de la meva propietat o el contingut de taules extretes d'internet.

10.1 Consultes web

1. ***Android Developer Reference***. [en línia, 2018] Disponible a : https://developer.android.com

2. ***Xampp Server*** . [en línia, 2018] Disponible a : https://www.apachefriends.org/download.html

3. ***Volley*** . [en línia, 2018] Disponible a : https://developer.android.com/training/volley/

4. ***SQLite3***. [en línia, 2018] Disponible a : https://packages.ubuntu.com/xenial/sqlite3

5. ***MySQL*** . [en línia, 2018] Disponible a : https://www.mysql.com/

6. ***SQLyog*** . [en línia, 2018] Disponible a : https://www.webyog.com/product/sqlyog

7. ***ObsProject*** . [en línia, 2018] Disponible a : https://obsproject.com/download

8. ***PHPMailer***. [en línia, 2018] Disponible a : https://github.com/PHPMailer/PHPMailer

9. ***ItextPDF***. [en línia, 2018] Disponible a : https://mvnrepository.com/artifact/com.itextpdf/itextg/5.5.10

10. ***Volley Github***. [en línia, 2018] Disponible a : https://github.com/google/volley/releases/tag/1.1.0

11. ***Password-hash*** [en línia, 2018] Disponible a : http://php.net/manual/en/function.password-hash.php

12. ***StackOverflow***. [en línia, 2018] Disponible a : https://stackoverflow.com/

13. ***Comunicació entre fragments SekthDroid***. [en línia, 2018] Disponible a : https://sekthdroid.wordpress.com/2013/02/02/comunicacion-entre-fragments-en-android/

14. ***Algoritme Bcrypt***. [en línia, 2018] Disponible a : https://en.wikipedia.org/wiki/Bcrypt

15. ***Async Task***. [en línia, 2018] Disponible a : https://jarroba.com/asynctask-en-android/

16. ***Tutorials Android***. [en línia, 2018] Disponible a : https://www.youtube.com/user/CristianDavidHenao/

17. ***Volley Singleton Example***. [en línia, 2018] Disponible a : https://github.com/CypressNorth/Volley-Singleton/blob/master/VolleySingleton.java

18. ***000webhost***. [en línia, 2018] Disponible a : https://www.000webhost.com/

19. ***SHA-1***. [en línia, 2018] Disponible a : https://es.wikipedia.org/wiki/Secure_Hash_Algorithm#SHA-1

20. ***MD5***. [en línia, 2018] Disponible a : https://en.wikipedia.org/wiki/MD5

21. ***PHP tutorial***. [en línia, 2018] Disponible a : https://www.w3schools.com/pHP/default.asp

22. ***Java***. [en línia, 2018] Disponible a : https://www.tutorialspoint.com/java/index.htm

10. 2 Imatges i taules extretes d'internet

23. ***Competència entre sistemes operatius***. [en línia, 2018] Disponible a : http://infographic.statista.com/normal/chartoftheday_8203_android_e_ios_dominan_mas_del_99por_ciento_del_mercado_n.jpg

24. ***Taula % usuaris per cada versió Android***. [en línia, 2018] Disponible a : https://www.muycomputer.com/2017/12/12/android-oreo-cuota-mercado/amp/

25. ***Logo Android Studio***. [en línia, 2018] Disponible a : https://commons.wikimedia.org/wiki/File:Android_Studio_icon.svg

26. ***Logo Xampp***. [en línia, 2018] Disponible a : https://i1.wp.com/www.betterhostreview.com/wp-content/uploads/2017/02/xampp-logo.jpg?resize=500%2C142

27. ***Logo 000webhost***. [en línia, 2018] Disponible a :
https://s3-eu-west-1.amazonaws.com/tpd/logos/47fcb9400000640005023f0b/0x0.png

28. ***Logo SQLite***. [en línia, 2018] Disponible a :
https://commons.wikimedia.org/wiki/File:SQLite370.svg

29. ***Logo MySQL***. [en línia, 2018] Disponible a :
https://seeklogo.com/vector-logo/96581/mysql
30. ***Logo SQLyog***. [en línia, 2018] Disponible a :
http://fullversionbae.blogspot.com/2017/09/sqlyog-ultimate-1243-full-version.html

31. ***Logo OBS Studio***. [en línia, 2018] Disponible a :
https://i1.wp.com/gameishard.gg/wp-content/uploads/2017/03/obs_new_icon_wide.png?resize=1280%2C640&ssl=1

32. ***Logo Twitch***. [en línia, 2018] Disponible a :
https://www.twitch.tv/p/assets/uploads/combologo_474x356.png

33. ***Logo PHPMailer***. [en línia, 2018] Disponible a :
https://avatars1.githubusercontent.com/u/3959702?s=400&v=4

34. ***Logo ItextPDF***. [en línia, 2018] Disponible a :
https://itextpdf.com/sites/default/files/ITSC-avatar.png

35. ***Logo Volley***. [en línia, 2018] Disponible a :
https://1.bp.blogspot.com/-euLsWEdhtdA/WJ7MCSBc8fI/AAAAAAAAAL8/LNwtC1WrNFwoMjb5xSoD82vmfNO6DPkGACLcB/s1600/android-volley-library.jpg

36. ***Logo PHP***. [en línia, 2018] Disponible a :
http://php.net/images/logos/new-php-logo.svg

37. ***Cicles de vida activitat***. [en línia, 2018] Disponible a :
https://developer.android.com/images/activity_lifecycle.png?hl=es-419

38. ***Cicles de vida fragment***. [en línia, 2018] Disponible a :
https://developer.android.com/images/fragment_lifecycle.png

39. ***Linear Layout***. [en línia, 2018] Disponible a :
https://www.tutorialspoint.com/android/images/liner.jpg

40. ***Relative Layout***. [en línia, 2018] Disponible a :
https://developer.android.com/images/ui/relativelayout.png

41. ***Característiques 000webhost***. [en línia, 2018] Disponible a :
https://www.000webhost.com/features

Annex A. Estudi del desenvolupament d'aplicacions Android

En aquest apartat s'analitzaran els elements utilitzats en el projecte. S'explicarà el funcionament d'activitats i fragments, on es podrà veure, els cicles de vida i la implementació de les interfícies gràfiques. Per concloure, s'explicaran quins són els elements que formen la IU.

Funcionament d'activitats i fragments

En l'estructura del treball, gairebé tots els arxius JAVA que es troben al projecte s'utilitzen per descriure com es comportaran les activitats i els fragments (pantalles) amb els estímuls de l'usuari.

Activitat

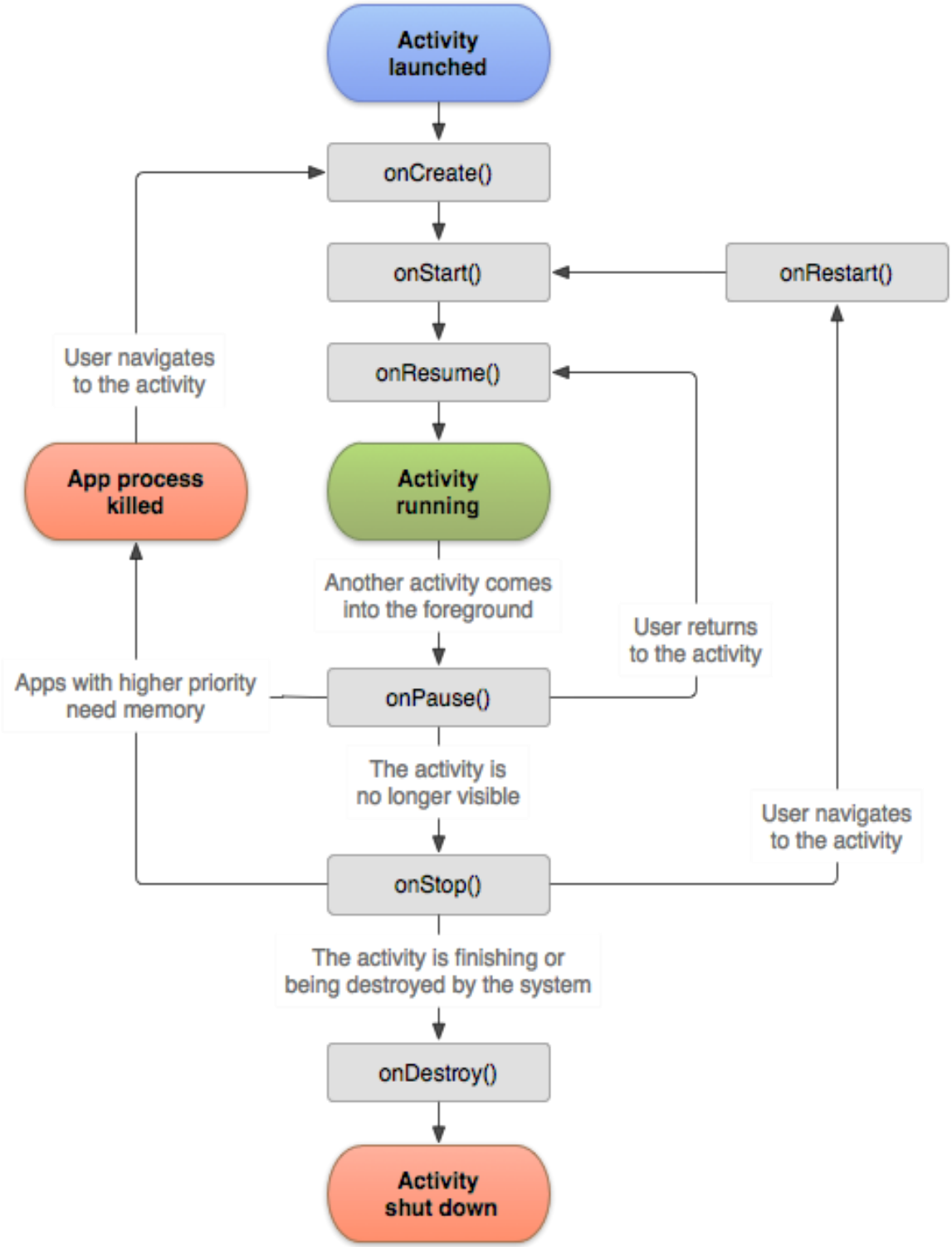
Una **activitat** és un component de l'aplicació que conté una pantalla amb la qual els usuaris poden interactuar, realitzar una acció, com per exemple, marcar un telèfon, fer una foto, enviar un missatge... A cada activitat, se li assigna un finestra, amb un disseny d'interfície d'usuari. Normalment, aquesta finestra ocupa tota la pantalla o queda flotant per sobre una altra.

En la majoria d'aplicacions, aquestes estan formades per múltiples activitats interconnectades/vinculades entre si de manera flexible. Una d'elles s'especifica com activitat principal, normalment acostuma a ser la que es presenta primer a l'usuari. Cada activitat és capaç d'iniciar una altra per poder realitzar diferents funcions. Quan es passa d'una activitat a un altre, l'anterior no es destrueix o és tenca, simplement s'emmagatzema a la pila pausada, on posteriorment, quan l'usuari prem el botó enrere, és capaç de recuperar-la.

Cada cop que es produeix un canvi en l'activitat, com per exemple, iniciar una de nova, es notifica el canvi mitjançant mètodes de *callback* (crear, aturar, reanudar o destruir) d'aquesta manera, es pot realitzar una tasca específica per a cada canvi d'estat.

Cal destacar el mètode **onCreate()** i **onPause().** El primer és un mètode que sempre s'ha d'implementar. S'utilitza en el moment de crear l'activitat i inicialitza tots els components gràfics. En el mètode has de cridar el **setContentView(XML id)** que rebrà com a paràmetre un *layout* en format XML definint així, el disseny de la interfície d'usuari.

Respecte a la segona, cada cop que s'abandona l'activitat es crida onPause(). Quan succeeix això, has de verificar quins canvis han de conservar-se o quins no, per si l'usuari torna a l'activitat. Les activitats tenen un cicle de vida que principalment es pot dividir en tres estats, executada, pausada i detinguda.



Il·lustració 68 : Cicle de vida de l'activitat. FONT: Externa[37]

En aquesta figura es poden veure tots el mètodes de *callback* que poden aparèixer en el cicle de vida de l'activitat. De cicles de vida, podríem destacar-ne tres diferents:

- **Cicle de vida complet**: Transcorre des de el mètode *onCreate()* i la crida a *onDestroy().* Tots els recursos que utilitzava l'activitat acaben alliberats.
- **Cicle de vida visible:** Transcorre des de la crida a *onStart()* fins a *onStop().* Durant aquest procés, l'usuari pot veure la pantalla i interactuar amb ella.
- **Cicle de vida en primer pla:** Transcorre entre la crida a *onResume()* i la crida a *onPause().* Durant aquest temps, l'activitat es troba en primer pla en la pantalla. Un cop surt del primer pla, es crida al mètode *onPause()* quan el dispositiu entra en suspensió o apareix un diàleg (finestra emergent que apareix sobreposant-se a la resta).

Implementar una interfície d'usuari:

La interfície d'usuari bé donada per una jerarquia de vistes, les quals són objectes derivats de la classe *View*. Cada vista controla un espai rectangular específic dins de la finestra de l'activitat i es l'espai on pot interactuar l'usuari.

Android, ofereix un conjunt d'elements per organitzar i dissenyar les IU. Els *widgets* són elements visuals e interactius que s'utilitzen en el disseny de les interfície d'usuari. Android disposa d'una gran varietat de *widgets* com podrien ser *ratingBars, seekbars*; Botons, *Layouts*, *Containers*... però, també pots dissenyar-ne de nous totalment personalitzats.

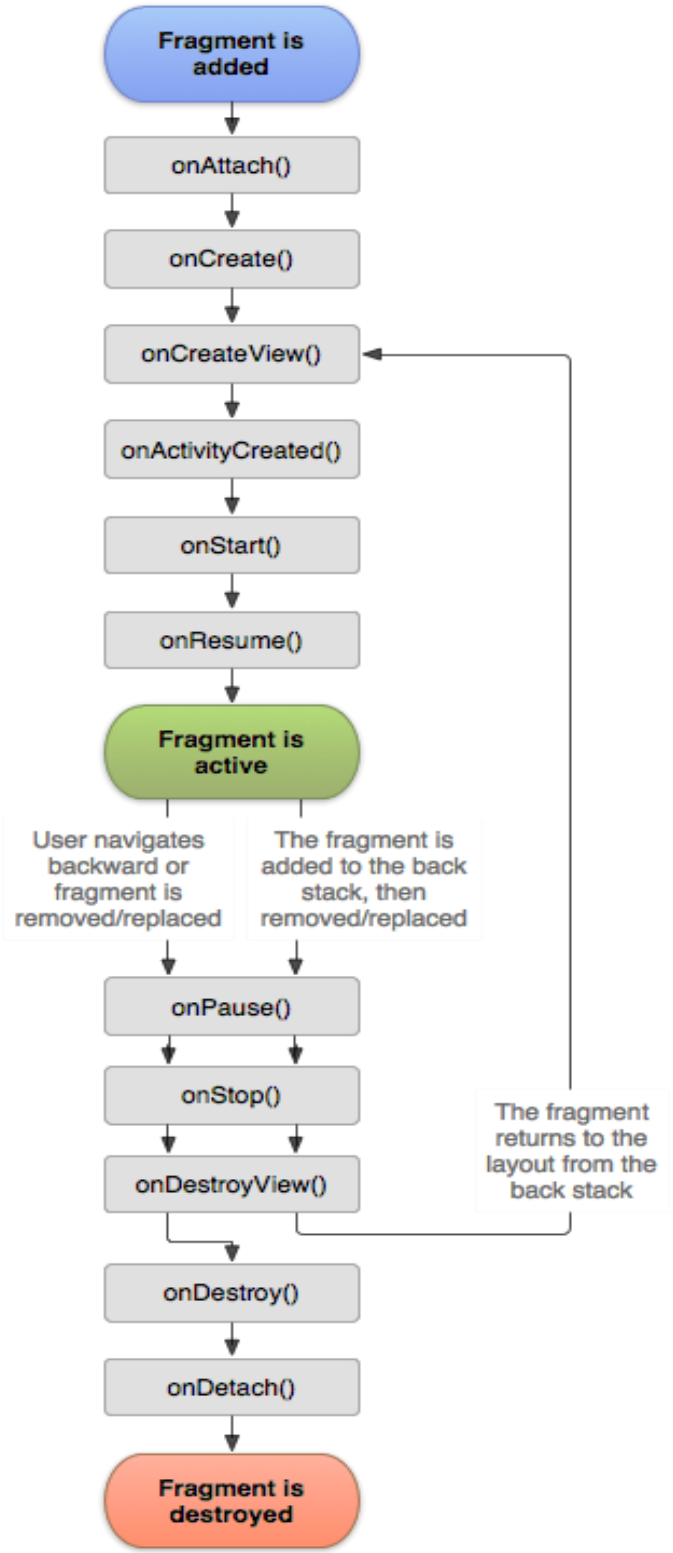
S'acostuma a tenir els dissenys en un arxiu XML dins de la carpeta res del projecte, amb l'objectiu de separar el codi font de la interfície d'usuari i el comportament amb Java. Amb la funció ***setContentView(XML id)*** podem obtenir el recurs XML del disseny.

A més a més, si l'activitat no està declarada en l'arxiu ***manifest*****,** no serà accessible pel sistema. En aquest arxiu, també s'haurà d'especificar quina és l'activitat principal utilitzant ***<intent-filter></intent-filter>*** i **els atributs *action.main i category.LAUNCHER,*** amb l'objectiu d'aclarir quina activitat s'inicialitzarà un cop es comenci a executar l'aplicació.

Fragment

Els fragments són una **part o comportament** d'una activitat. Dins d'una activitat és poden combinar múltiples fragments. El fragment sempre ha d'estar integrat en una activitat i el seu cicle de vida es veurà directament afectat pel cicle de vida de l'activitat amfitriona. Mentre que l'activitat està en execució, es pot manipular els fragments que la componguin.

El cicle de vida d'un fragment es podrà veure a la següent pàgina, sempre que la seva activitat amfitriona estigui en execució:



Il·lustració 69 : Cicle de vida del fragment. FONT: Externa[38]

Dels mètodes exposats en el gràfic podríem destacar-ne ***onCreateView()***, en aquest es on es faran les inicialitzacions gràfiques. En cas que no tingui disseny gràfic, pot retornar un *null*.

Implementar una interfície d'usuari:

Per tal de poder obtenir la interfície gràfica que correspon al fragment, s'ha de retornar l'objecte resultant del mètode ***inflate().*** Aquest mètode permet extreure una nova jerarquia de vistes gràcies al recurs XML. La funció necessita 3 paràmetres:

- ID: recurs del disseny a utilitzar.
- *ViewGroup*: rep com a paràmetre el *container*, el qual, té la funció d'especificar al sistema sobre quina vista s'ha d'afegir el disseny del fragment.
- Booleà: indica si el disseny s'ha d'annexar al ***ViewGroup*** (Vista especial que permet inserir-li altres vistes més petites anomenades fills), en aquest cas sempre serà *false*.

Els fragments no sempre estan associats a un IU, per aquest motiu, es pot afegir un fragment des de l'activitat amb un simple *add*(Fragment, *String*) proporcionant una etiqueta única com a *String*.

Gestió dels fragments

Sempre que es vulguin administrar, s'ha de fer servir el ***FragmentManager*****.** Aquest ens dóna la possibilitat de realitzar diverses funcions interessants:

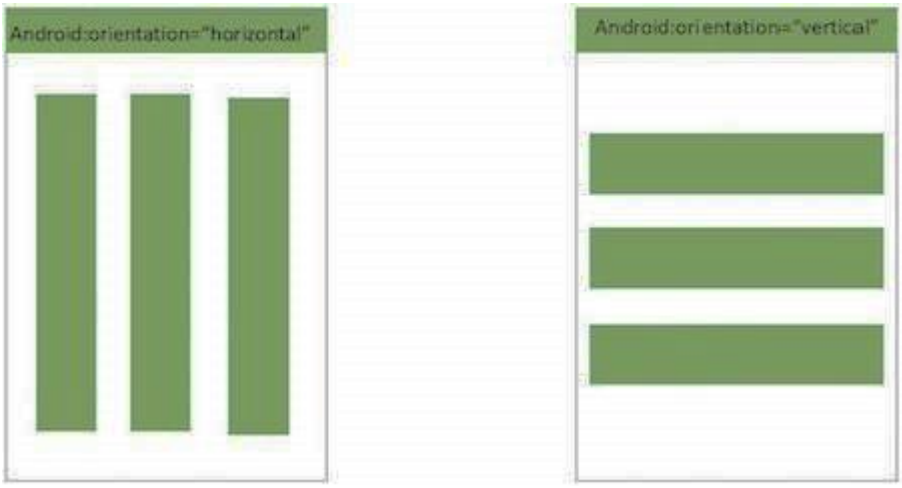
- ***findFragmentById():*** Obtindrà fragments que ja existeixen en l'activitat i tenen interfície gràfica.
- ***findFragmentByTag():*** Obtindrà el fragment que pot o no, tenir UI.
- ***popBackStack():*** Desempila un fragment per tal d'executar-lo.
- ***addBackToStack():*** Afegeix a la pila un fragment.

El ***fragmentManager*** a més a més permet l'ús de transicions. Aquest pot obtenir una instància de la classe ***FragmentTransaction***. Amb aquesta classe mitjançant les funcions de *remove(), add() i replace()* es pot gestionar tot tipus de transicions entre fragments. Amb la comanda *commit(),* s'apliquen els canvis i sempre s'ha de cridar com a últim mètode de la transició.
Cal destacar, que si es realitza una transició sense cridar al mètode *addToBackStack()*, no es podrà recuperar el fragment anterior perquè aquest es destrueix, però, si s'utilitza, el fragment es queda a la pila en l'estat de pausat.

Widgets, Botons, *Containers*, Texts i *Layouts* que conté l'aplicació desenvolupada

Al llarg del treball, s'utilitza varis *Layouts*. Principalment la combinació de *ScrollView+LinearLayout* en la majoria de fragments. Tot i així, en apartats on necessiten un disseny visual molt específic, s'ha utilitzat un *RelativeLayout*.

LinearLayout: És un grup de vista que "*encara*" o ordena tots els seus components en una única direcció, horitzontal o vertical. És pot especificar la direcció amb l'atribut ***android:orientation***.



Il·lustració 70 : Linear Layout. FONT: Externa[39]

***Relative Layout*:** És un *layout* que permet especificar la ubicació dels objectes secundaris en funció de la posició relativa d'altres objectes secundaris que integren el *layout*. Per exemple, es pot especificar que l'objecte B, estigui a 10 dp (mesura de densitat independent de píxels) per sota de l'objecte A.

Il·lustració 71 : Relative Layout. FONT: Externa[40]

ScrollVIew*:** És un ***View Group que permet contindré més elements que la mida de la mateixa pantalla. Aquest només permet tenir un fill directe, que ha de ser forçosament, un *LinearLayout*. Cal destacar que dins d'aquest, pot haver-hi altres

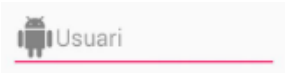
layouts. La principal característica és que apareix una barra lateral que permet desplaçar amunt i avall la pantalla com un navegador d'internet. Si vol tenir suport de desplaçament horitzontal, s'ha d'utilitzar ***HorizontalScrollView.***

Els principals components que integren els *layouts* utilitzats són els següents:

CardView: Hereta mètodes del *FrameLayout,* es semblant a un botó, però aquest *widget* té costats rodons, una elevació i ombra pels contorns.

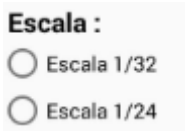
Il·lustració 72 : CardView de l'aplicació. FONT: Pròpia

EditText: **IU** que permet inserta i modificar text. Quan es crea l'objecte, s'ha d'especificar quin format de text accepta: contrasenyes, text pla, correu... S'acostuma a mostrar text per defecte amb l'atribut ***android:hint***.



Il·lustració 73 : EditText. FONT: Pròpia

RadioButtons*:* Permet a l'usuari seleccionar una opció de les disponibles. S'utilitza en casos on les opcions són exclusives mútuament. Per exemple, alhora de seleccionar en quina escala vols participar. Un conjunt de *RadioButtons* formen un ***RadioGroup***.



Il·lustració 74 : RadioButtons. FONT: Pròpia

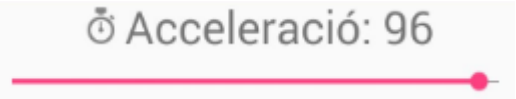
Spinners: Proporcionen un menú desplegable amb varies opcions a triar. Sense haver seleccionat cap opció, mostra un valor determinat. Normalment s'acostumen a implementar mitjançant un ***SpinnerAdapter***, el qual, li passa com a paràmetre un document de la carpeta res, on estan llistades les opcions.

Il·lustració 75 : Spinner. FONT: Pròpia

Text i botons: Els botons consisteixen en un text o un icona (o ambdues coses) tal que un cop són premuts, s'inicia un esdeveniment. Els ***TextView*** ofereixen la possibilitat de mostrar els textos a l'usuari.

Il·lustració 76 : Botó. FONT: Pròpia

***SeekBar*:** Hereta mètodes de la classe *progressBar*, amb la diferència que permet arrossegar la barra en una posició determinada, la qual, retorna un *int* de valors 0 a 100.



Il·lustració 77 : Seekbar. FONT: Pròpia

Per poder "escoltar" els clicks que realitza l'usuari s'acostuma a implementar dins del codi Java un ***setOnClickListener() ó setOnItemSelected()*** ... Depenent del *widget* per tal de processar l'esdeveniment necessari.

Annex B. Estudi de les injeccions SQL

Els *scripts* amb PHP servits pel servidor permeten gestionar la informació que envia o rep l'aplicació. Com que Volley no és capaç de suportar MySQL directament, l'ús de PHP és imprescindible per realitzar les connexions a la base de dades.
Els *scripts* gestionen peticions POST/GET provinents del terminal Anroid. Un cop es reben els paràmetres, mitjançant una connexió a la base de dades es pot realitzar la consulta, inserció, eliminació o actualització de la informació.

PHP, disposa de funcions que permeten accedir a la base de dades amb MySQL de manera senzilla i eficaç. Fins a la versió **5 de PHP**, s'utilitzava les funcions amb extensió **MySQL,** però, a partir d'aquesta és recomana utilitzar les de **MySQLi** perquè suposa una millora important vers l'extensió anterior:

- Interfície dual. Suporta la programació procedimental i l'orientada a objectes.
- Suporta sentències parametritzades.
- MySQLi: extensió millorada en seguretat i *debuggejada*.
- MySQLi: suporta transaccions ACID a nivell d'API.
- Dóna solucions a possibles injeccions SQL.
- Permet múltiples consultes amb una simple expressió com ***multi_query()***

Com evitar **injeccions de codi SQL a PHP**?

Les injeccions SQL és una via d'**infiltració** de codi intrús que utilitza un error en el programari d'una aplicació, a nivell de validació dels *inputs,* per poder realitzar consultes a la base de dades. La vulnerabilitat naix en l'incorrecte filtratge de les variables utilitzades en un programari o API que executa sentències SQL.

Existeixen dos procediments per evitar-les i no tenir errors de seguretat:

Escapament de variables:

mysqli_real_escape_string: Permet transformar una cadena, escapant els caràcters especials en SQL pel seu equivalent textual, d'aquesta manera, s'interpreta el contingut de la variable com si fos text. Cal destacar que la funció ***~~mysqli_escape_string~~***, que s'utilitzava fa 10 anys aproximadament, no era segura, però, PHP en comptes de modificar-la i actualitzar-la, ha hagut de mantenir-la per problemes de compatibilitat per això, ha afegit **REAL** al nom de la funció nova totalment *debuggejada*.

***IntVal*:** Converteix un valor escalar en un enter. En cas d'error retorna un 0.

Amb aquestes dues funcions es saneja les variables introduïdes per l'usuari. La primera s'utilitza amb *strings* mentre que la segona amb paràmetres que han de ser enters.

Sentència parametritzada SQL (*prepared statement*):

1. Preparar. La sentència es transforma en una plantilla i és enviada a la base de dades . Els valors que queden per especificar s'anomenen paràmetres o *bind variables*.

2. El gestor de la base de dades compila i realitza una optimització en la sentència que s'ha utilitzat com a plantilla i emmagatzema el resultat sense executar-lo.

3. Executar. L'aplicació substitueix (funció *blind*) els paràmetres i s'executa la sentència. L'API pot executar la sentència tantes vegades com valors diferents es tinguin.

La gran diferència que suposa això respecte executar sentències SQL directament, és que els paràmetres es transmeten amb un protocol diferent i aquests estan perfectament escapats. Si la sentència a executar no prové d'un element extern, mai es produirà una injecció.

Funcions disponibles per a poder crear sentències parametritzades:

- ***mysqli_prepare*:** Àlies de la funció *mysqli_stmt_prepare()*.Prepara una sentència SQL per a la seva execució.
- ***mysqli_bind_param*:** Actua com a àlies de la funció *mysqli_stmt_bind_param().* Enllaça/relaciona i saneja les variables que es reben com a paràmetres en la sentència parametritzada.
- ***mysqli_execute*:** Actua com a àlies de la funció *mysqli_stmt_execute()*. Executa una funció preparada.

Altres funcions disponibles:

- ***mysqli_connect*:** Obre una connexió al servidor MySQL. Rep com a paràmetres el nom del servidor, el nom d'usuari, la contrasenya per la connexió i per últim la base de dades a manipular.
- ***mysqli_affected_rows*:** Obté el nombre de files afectades en la última consulta *INSERT, UPDATE, REPLACE o DELETE*.
- ***mysqli_query***: Executa una sentència SQL.
- ***mysqli_fetch_array*:** Obté un *array* amb la forma del resultat de la consulta. Permet accedir a les dades.

Cal destacar la funció ***isset(),*** capaç de determinar si una variable esta definida i no és nul·la. Aquesta té sentit utilitzar-la en la recepció de paràmetres amb PHP, on si la petició no s'envien tots els camps necessaris, simplement es refusa la connexió a la base de dades.

Annex C. Manual de l'usuari

Al llarg de l'aplicació l'usuari pot recórrer diverses pantalles amb la capacitat d'interactuar i realitzar accions. La majoria de pantalles estan destinades a funcions específiques o merament informatives.

Per fer més fàcil les indicacions, **s'afegiran petits nombres dins d'un cercle de color vermell** per tal de fer entendre millor a quin element visual s'està referint el manual.

SPLASH SCREEN:
La primera pantalla que veurà la persona que arranqui l'aplicació serà la següent:

Il·lustració 78 : Splash Screen. FONT: Pròpia

Aquesta pantalla apareix durant **3 segons**. Serà visible sempre que s'obri l'aplicació. Posteriorment, dóna pas al *login*.

LOG IN:
En aquesta pantalla l'usuari pot realitzar 6 accions diferents:

Il·lustració 79 : Login. FONT: Pròpia

- Prement el botó **(1)** es desplega un menú amb un llistat de pantalles de l'aplicació, la qual cosa, permet desplaçar-te ràpidament entre elles.
- Obre **(2)** un petit menú desplegable on es troba l'opció de *desloggejar-se* i una petita recerca d'informació sobre el desenvolupador de l'aplicació.
- Permet **(3)** accedir a la pantalla on es pot recuperar la informació del compte.
- Un cop es *fa click* **(4),** realitza una petició de connexió al servidor amb les dades introduïdes en els camps d'usuari i contrasenya. Si els camps són correctes, s'inicia la sessió. En cas contrari, s'informa a l'usuari.
- Permet **(5)** accedir a la pantalla de registrar usuari.
- Permet **(6)** accedir al menú sense haver iniciat sessió.

Cal destacar que **no es necessari *loggejar-se*** cada vegada que s'obra l'aplicació. Sempre que no s'hagi tancat sessió anteriorment, el compte seguirà obert.

L'opció **1 i 2** són accessibles des de qualsevol pantalla, per tant, no s'explicaran de nou al llarg del manual.

REGISTRE:
Aquesta pantalla permet registrar un nou usuari a l'aplicació. S'han d'introduir **obligatòriament** els camps d'usuari, contrasenya, repetir contrasenya i el correu. El nom d'**usuari** pot tenir com a màxim **20 caràcters**. El **correu** com a **màxim 60 caràcters**. La **contrasenya** a introduir serà **de 72 caràcters com a màxim**. Si l'usuari incompleix algun requisit de llargària o forma, se l'avisarà amb *un toast message*.

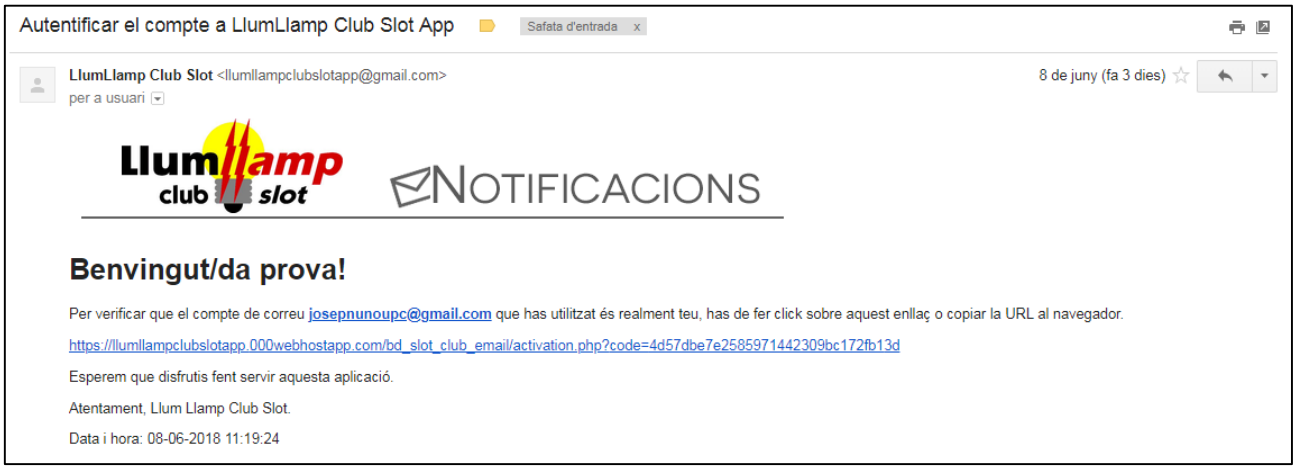
Il·lustració 80 : Registre. FONT: Pròpia

Per aconseguir crear exitosament el nou compte, s'ha d'acceptar les condicions d'ús. Aquestes estan servides i visibles per a tothom en el servei web de 000.webhostapp.com en format PDF, el qual podran descarregar-se fent *click* en "Veure les condicions d'ús".

Punts a destacar:

- S'ha de marcar la casella **(1)** per poder crear de manera exitosa un compte nou.
- Un cop es faci *click* **(2)** en registre!, es crearà un compte nou i s'enviarà un correu a l'e-mail especificat. L'usuari haurà de verificar el correu *clickant* l'enllaç que trobarà al correu o copiant l'*URL* al navegador, per tal de poder fer servir el 100% de les funcionalitats. En cas de no acceptar-lo**, no podrà realitzar cap inscripció a les curses.**

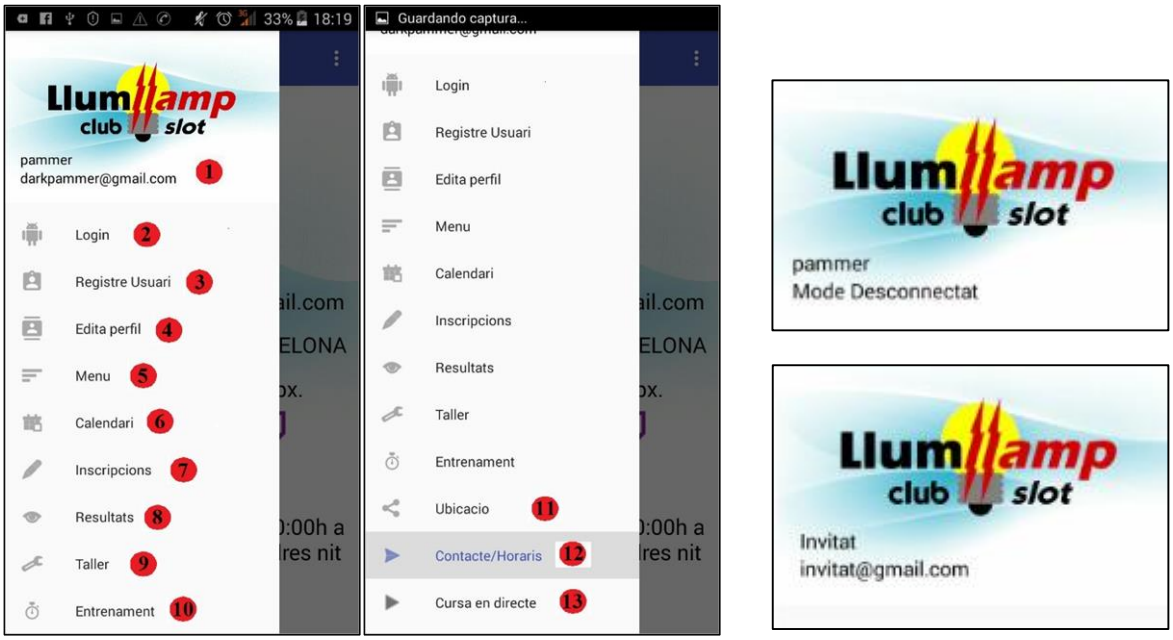
El missatge que rebrà al correu serà un amb aquest format:



Il·lustració 81 : Correu d'activació. FONT: Pròpia

Quan l'usuari fa *click* en l'enllaç, **s'activa** el compte d'usuari. El correu que utilitzarà serà el lloc on s'enviarà **els resguards / tiquets de les inscripcions de les curses, gestions de seguretat...** Per tant, ha de ser un correu amb una contrasenya molt segura i de la seva propietat.

MENÚ DESPLEGABLE:
Aquesta eina facilitarà el canvi de pantalles a través de l'aplicació:



Il·lustració 82 : Conjunt imatges Navigation Drawer. FONT: Pròpia

D'aquest menú, cal destacar la capçalera **(1).** Aquesta està formada pel nom de l'usuari i el correu. En cas d'estar en mode desconnectat, apareix "**Mode Desconnectat**" en comptes del correu. Si l'usuari no està *loggejat*, apareix com **invitat**.

Les opcions 4,7,10 necessiten les credencials de l'usuari, per tant, has d'iniciar sessió per accedir-hi. Tot i així, **l'opció 10**, permet l'accés en **mode desconnectat**. Les funcionalitats restants són accessibles per a qualsevol tipus d'usuari. Respecte a **les opcions 11, 12 i 13** només són accessibles des de el *navigation drawer.*

11. Ubicació: Automàticament s'obre Google Maps amb la ubicació ja establerta del club perquè l'usuari conegui com arribar de la forma més ràpida possible.

Il·lustració 83 : Google Maps Ubicació. FONT: Pròpia

12. Pantalla on es troba informació sobre l'horari i els mitjans disponibles per contactar amb el club.

Il·lustració 84 : Contacte i horaris. FONT: Pròpia

13. Cursa en directe no té una pantalla pròpia dins de l'aplicació. Un cop fas *click*, automàticament s'obre el navegador amb un enllaç que et porta al canal de **Twitch** de Llum Llamp. Aquesta plataforma permet veure les retransmissions de les curses al llarg del cap de setmana.

MENÚ:
Conté principalment les funcions essencials de l'aplicació. Permet accedir al calendari, taller, inscripcions, entrenament, resultats i al teu perfil.

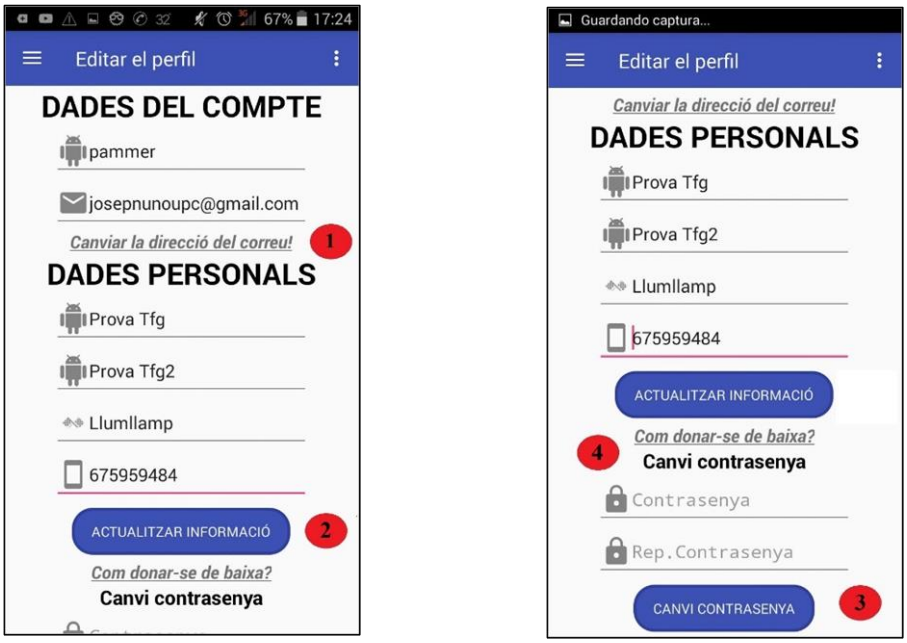
Il·lustració 85 : Menú principal. FONT: Pròpia

La pantalla està formada per 6 *cardviews* amb un o varies pantalles associades:

1. Calendari. Mostra la informació de les curses al llarg de l'any.
2. El taller permet realitzar la fitxa tècnica del vehicle.
3. Permet consultar els inscrits a una cursa i accedir posteriorment a la pantalla d'inscripcions.
4. Permet accedir al llistat de configuracions de comandaments i gestionar-los.
5. Accés a la pantalla on es pot comprovar els resultats de les curses.
6. Permet editar el teu perfil.

Per accedir a les **opcions 3,4,6** cal estar **loggejat** . Cal destacar però, **que l'opció 4**, es accessible mitjançant el **mode desconnectat**. D'aquesta manera, es pot seguir gestionant les configuracions si no tens connexió de dades.

EDITAR PERFIL:
En aquesta pantalla l'usuari editarà les seves dades. Cal afegir, que l'únic paràmetre que **no podrà editar serà el correu**. Per modificar aquest camp, s'ha de seguir una sèrie de passos que s'explicaran posteriorment en la pantalla de canvi de correu.



Il·lustració 86 : Conjunt imatges Editar Perfil. FONT: Pròpia

L'opció número 1 dóna pas a la pantalla canvi de correu. **L'opció 2**, permet actualitzar les dades personals de l'usuari i el *nick* del compte. El **nom** pot tenir un **màxim de 20 caràcters**, el **cognom 20**, **l'escuderia 25** i **telèfon mòbil 15**.

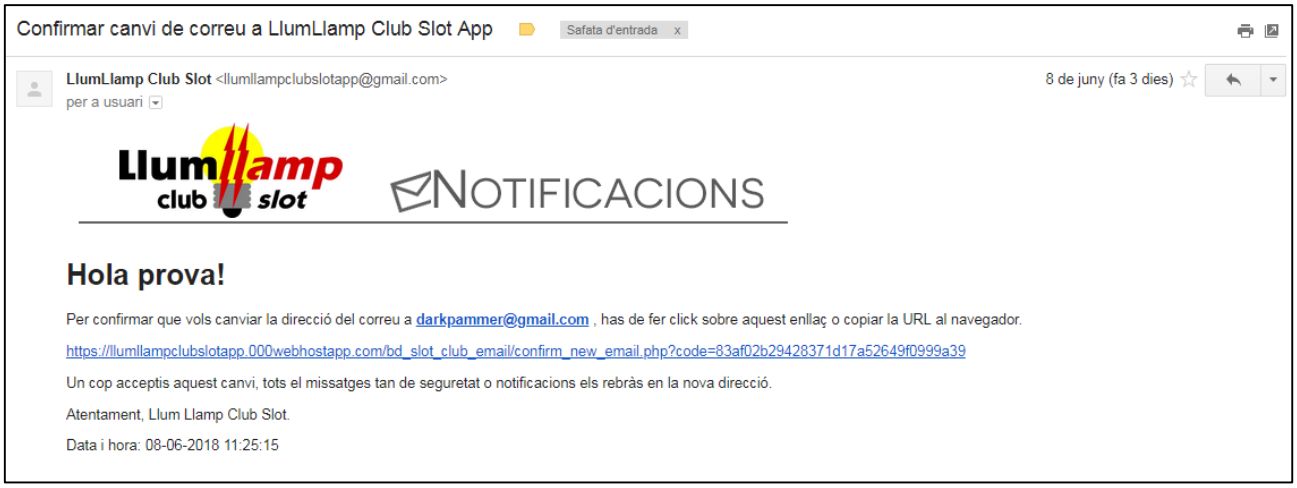
També, es troba disponible l'opció de canviar la contrasenya **(3) sempre que tingui com a màxim 72 caràcters i com a mínim 8.** Si l'usuari incompleix algun requisit de llargària o forma, se l'avisarà amb un *toast message*. **L'opció 4**, informa com l'usuari pot donar-se de baixa.

CANVI DE CORREU:
Aquesta pantalla només **és accessible des d'Editar Perfil**. Permet canviar l'adreça de correu (**llargada màxima 60 caràcters**) seguint uns passos detallats a continuació:

Il·lustració 87 : Gestió del correu. FONT: Pròpia

Un cop es fa *click* al botó **(1),** automàticament s'envia un correu a l'e-mail antic per tal de verificar que és vol canviar la direcció.

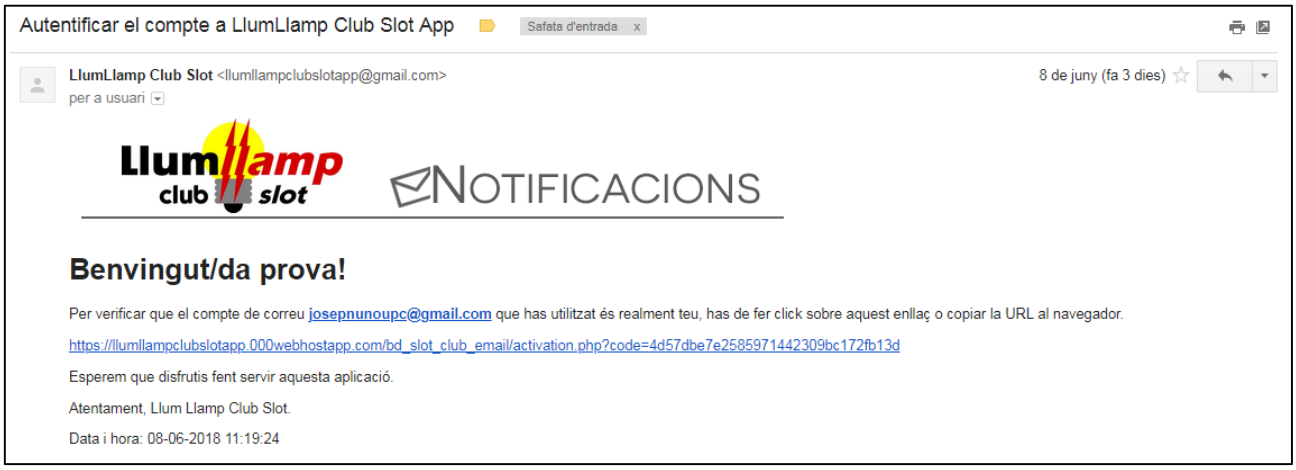
L'usuari rebrà un missatge d'aquest estil:



Il·lustració 88 : Confirmació de canvi correu. FONT: Pròpia

En aquest missatge s'indica explícitament, que si es fa *click*· en l'enllaç. S'assignarà un nou correu al compte de LlumLlamp. **Un cop es canvia la direcció del correu, totes les gestions de seguretat i resguards d'inscripcions arribaran al nou correu. No serà possible recuperar-lo.**

Posteriorment, s'envia un correu de verificació al nou e-mail per activar-lo.



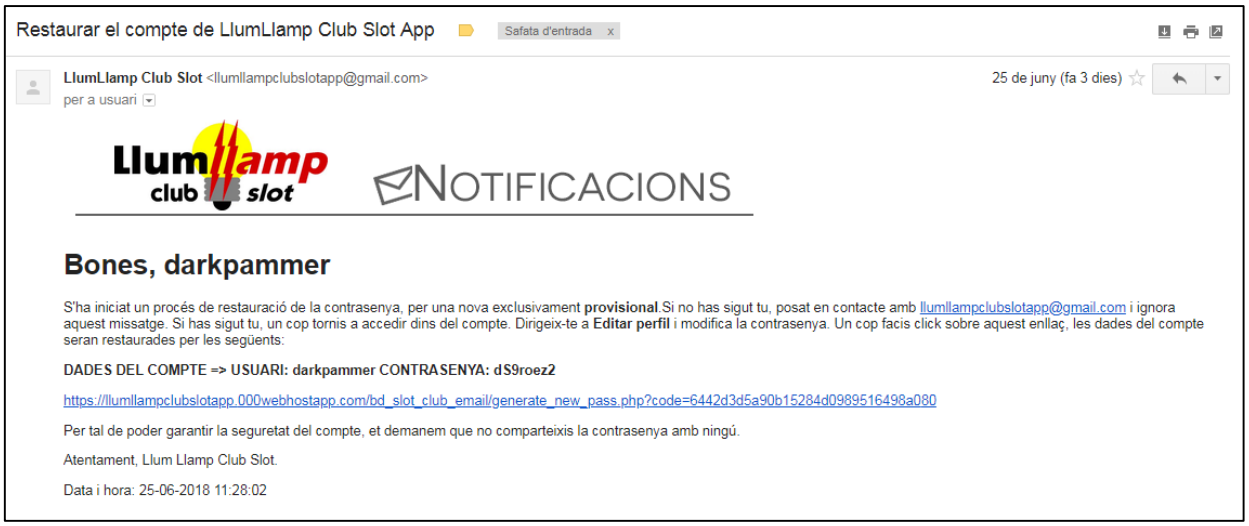
Il·lustració 89 : Correu de verificació. FONT: Pròpia

El mateix format de correu que es rep quan es **registra un usuari**.

RECUPERACIÓ DEL COMPTE:
Sempre que l'usuari perdi la contrasenya, el nom d'usuari o simplement algú hagi modificat el seu compte d'usuari, es pot recuperar accedit des de el Login fent *click* a l'opció de ***No pots accedir al compte?***

Il·lustració 90 : Recuperació del compte. FONT: Pròpia

Un cop es fa *click* **(1)** al botó, s'envia un correu a l'e-mail del compte amb un missatge que es veurà a continuació:



Il·lustració 91 : Correu recuperació del compte. FONT: Pròpia

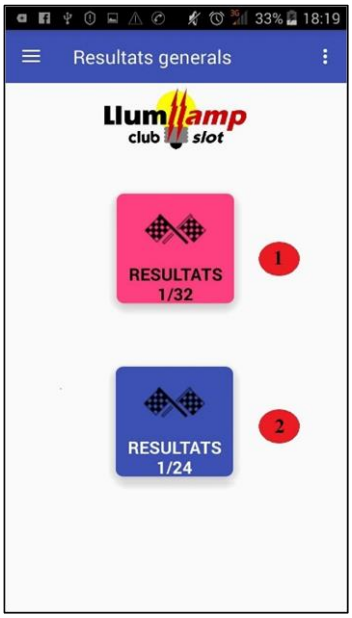
Per tant, només l'usuari **propietari del correu i del compte de LlumLlamp** podrà generar les noves credencials. La contrasenya per l'usuari s'actualitzarà amb el *click* a l'enllaç. Es pot enviar **un màxim de 3 peticions** de restauració per dia. D'aquesta manera s'evita l'*spam*.

CALENDARI:
Tal com indica el nom, aquesta pantalla serveix per trobar la informació de totes les curses que es realitzaran al club.

Il·lustració 92 : Calendari. FONT: Pròpia

L'usuari ha de fer *click* al *spinner* **(1)** on trobarà un menú desplegable amb tots els **mesos** de l'any. D'aquesta manera podrà **seleccionar quin mes vol consultar.**

RESULTATS:
Aproximadament un cop al mes es realitzen curses al club. Aquesta pantalla serveix per accedir de manera ràpida a la informació disponible a la pàgina web.



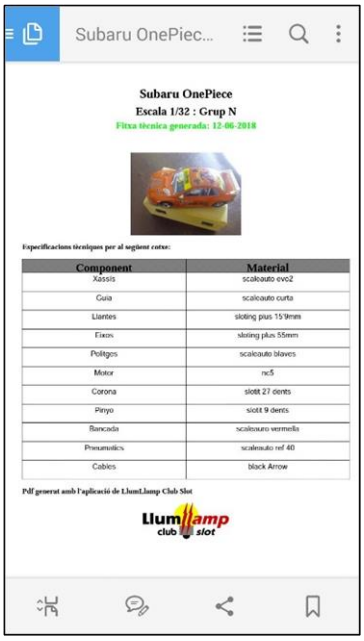
Il·lustració 93 : Resultats generals. FONT: Pròpia

L'opció 1, obre el navegador i automàticament es connecta a la direcció on es troba penjats els resultats de l'escala 1/32. Mentre que la segona **(2)**, als resultats de l'escala 1/24.

TALLER:
En aquesta pantalla accessible per a qualsevol usuari permet obtenir una fitxa tècnica del vehicle. L'objectiu d'aquesta funció **és merament informatiu**. Moltes vegades en el món de l'Slot, companys d'equip demanen informació sobre el material que porta el cotxe, per tant, una bona opció és generar una fitxa tècnica en format PDF per poder-la compartir utilitzant les xarxes socials.

Il·lustració 94 : Gestió cotxe. FONT: Pròpia

Aquesta pantalla permet omplir un conjunt de paràmetres que es troben dins d'un llistat de material bàsic. A més a més, es pot fer especificar el nom de la configuració, escala, categoria... Com a extra, utilitzant la càmera **(2)** permet fotografiar el vehicle en qüestió. Un cop els paràmetres estan complerts, l'usuari pot generar un PDF **(1).** El nom d'aquest document vindrà donat pel **nom de la configuració.** El document generat tindrà aquesta forma:

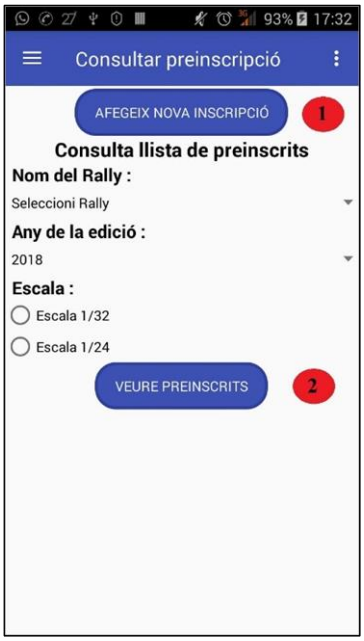


Il·lustració 95 : Document pdf. FONT: Pròpia

Aquest document està compost principalment per un títol i subtítol obtingut del nom de la configuració, escala, categoria i la data. Conté la imatge del vehicle i una taula on apareixen els valors especificats per a cada material.

INSCRIPCIONS:

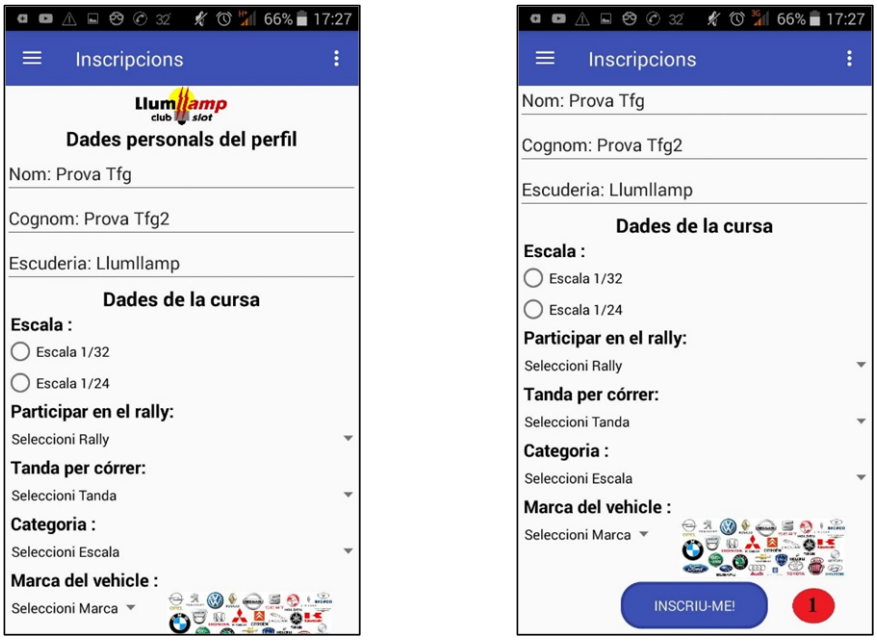
Un dels principals objectius de l'aplicació era permetre a l'usuari consultar els preinscrits a la cursa i a més a més, poder-se inscriure. En aquesta primera pantalla està la funció de consultar quins seran els pilots que participaran en la cursa.



Il·lustració 96 : Consulta preinscrits. FONT: Pròpia

Un cop l'usuari hagi seleccionat quin *rally*, edició i escala vol consultar. Fent *click* **(2),** podrà veure els participants. D'aquesta manera, el pilot podrà conèixer amb quins rivals competirà el cap de setmana.

En cas d'utilitzar **l'opció 1**, podrà generar una sol·licitud d'inscripció donant-se pas a una nova pantalla.



Il·lustració 97 : Conjunt imatges Inscripcions. FONT: Pròpia

En aquesta pantalla es pot gestionar els paràmetres d'inscripció. Les dades personals venen determinades pels valors introduïts en **Editar Perfil. Per tant, en cas de voler modificar algun paràmetre com el nom, cognom o escuderia l'usuari haurà d'anar a Editar Perfil mitjançant el menú desplegable o prement *back()* dues vegades i passant a través del menú.**

Quan un usuari vol inscriure's ha de determinar l'escala en la que vol participar, el *rally*, la tanda en la qual correrà, la categoria i la marca del vehicle. Per facilitar la inscripció, en cada paràmetre esmentat anteriorment, l'usuari haurà de fer *click* en el *spinner* corresponent on trobarà les opcions establertes. Un cop completat tots els paràmetres, només haurà de prémer el botó de **l'opció 1.**

Automàticament rebrà un correu amb tota la informació de la inscripció generada prèviament:

Il·lustració 98 : Resguard inscripcions. FONT: Pròpia

Aquest correu serveix com a justificant si sorgeix algun problema alhora d'organitzar la cursa.

ENTRENAMENT:
Aquest conjunt de pantalles tenen la intenció de permetre ajustar els comandaments de l'usuari quan corren amb diferents cotxes. D'aquesta manera es pot tenir un registre de quina acceleració, freno, *anti-spin*, voltatge utilitzes en cada cas.

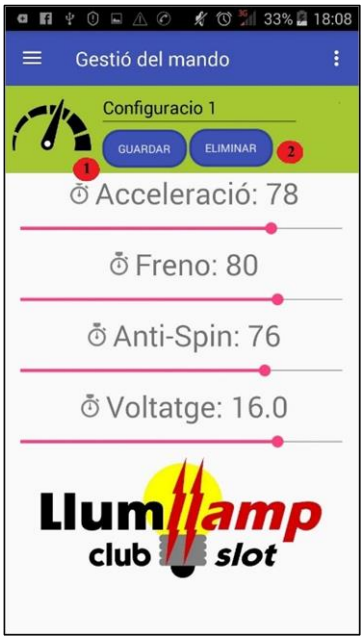
Il·lustració 99 : Llistat Configuracions. FONT: Pròpia

La primera pantalla mostra el conjunt de configuracions emmagatzemades per l'usuari. Un cop es fa *click* en alguna d'elles **(2,3,4),** es pot editar la informació individualment.

El nombre de configuracions està limitat per l'espai a memòria disponible en el telèfon, per tant, podem dir que pot arribar a tenir **fins a N on N és un nombre molt gran** (depèn de les prestacions del telèfon mòbil o tablet).

En cas que es vulgui afegir una nova configuració, s'haurà de fer *click* en **l'opció 1.**

Quan es vol editar una configuració o afegir-ne una de nova, es dóna pas a una pantalla d'aquest format:



Il·lustració 100 : Gestió del mando. FONT: Pròpia

Aquesta pantalla permet ajustar de manera simple tots els paràmetres d'acceleració, freno, *anti-spin* i voltatge mitjançant *seekbars*. Els **3 primers** esmenats recorren un interval des **de 0 fins a 100**. Pel que fa al **voltatge**, té precisió decimal **des de 0 fins a 20**.

LOG OUT I INFORMACIÓ:

Per poder tancar sessió **(1)** serà necessari adreçar-se al petit menú d'opcions a la part superior de la dreta de *toolbar.* **L'opció 2** porta a la pantalla d'informació sobre el club, el desenvolupador i informació sobre ser soci.

Il·lustració 101 : Menú desplegable Toolbar. FONT: Pròpia

Il·lustració 102 : Pantalla d'Informació. FONT: Pròpia

Annex D. *Streaming* de la competició.

Durant la cursa d'Slot, els pilots passen per 5 trams diferents. En cada tram obtenen un temps que s'anirà acumulant fins a completar els 5. Aquestes dades obtingudes s'introdueixen dins d'un programa que es diu **DS-Rally Control V.2**.

Il·lustració 103 : DS-rally control V.2. FONT: Pròpia

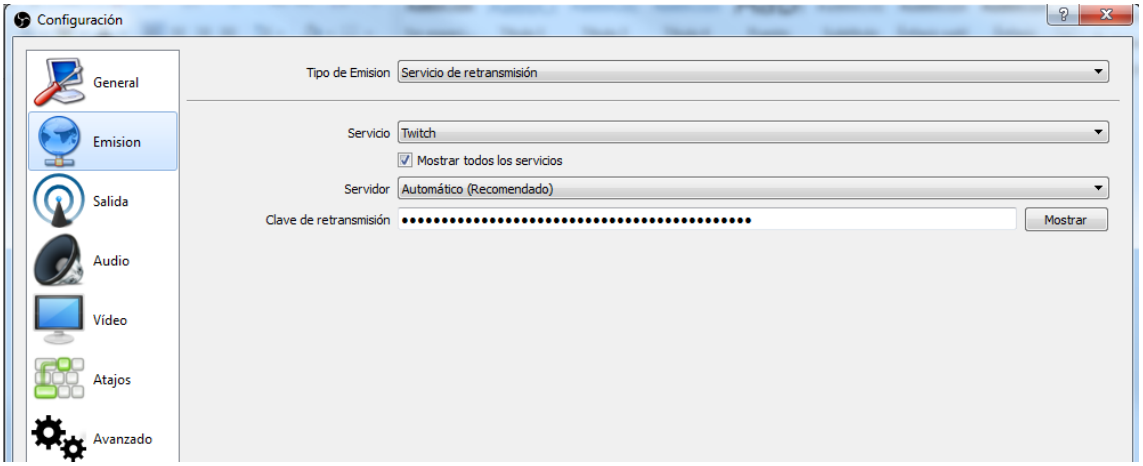
Un aspecte interessant era aconseguir capturar el programa i retransmetre'l en directe per tal de que tothom pugues veure com avança la cursa.

Per poder retransmetre en directe les curses s'utilitza l'eina d'**OBS Studio i Twitch.**

Primerament s'ha creat un compte a **Twitch** per a Llum Llamp. Cada compte té una clau de retransmissió exclusiva pel canal i totalment única. Aquesta clau permet identificar per a quin canal de Twitch és la transmissió, provinent d'un programa extern de retransmissió com pot ser Obs Studio, Xsplit, Bebo...

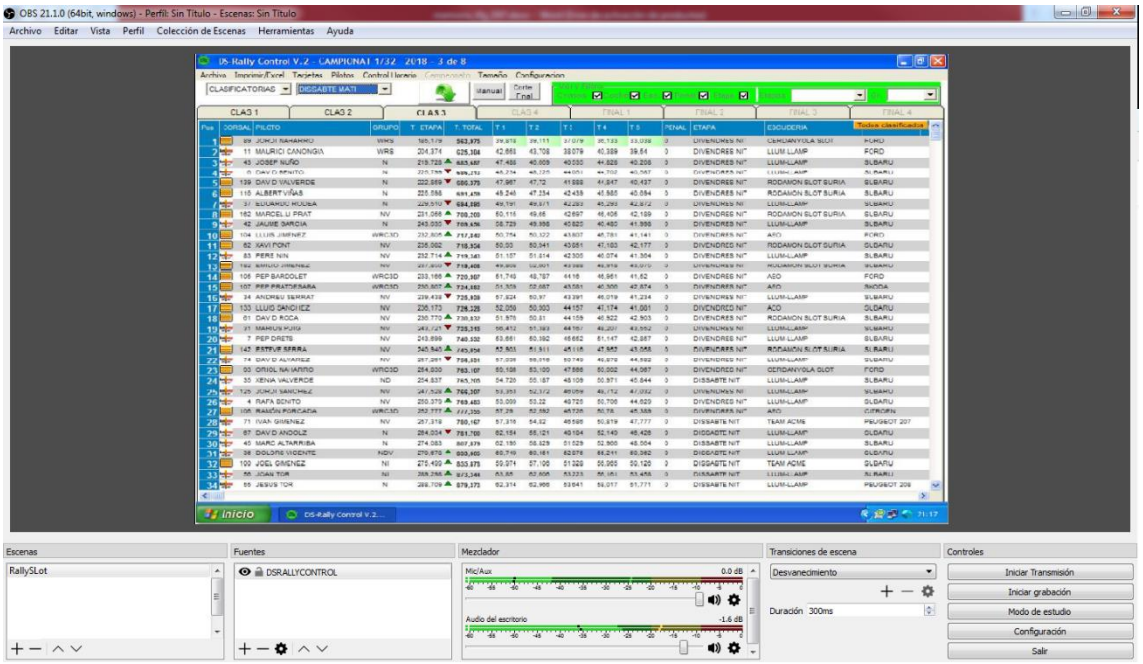
Il·lustració 104 : Clau de retransmissió Twitch. FONT: Pròpia

Per tant, el primer pas dins de la plataforma OBS Studio es especificar la clau de retransmissió que ens ha proporcionat Twitch:



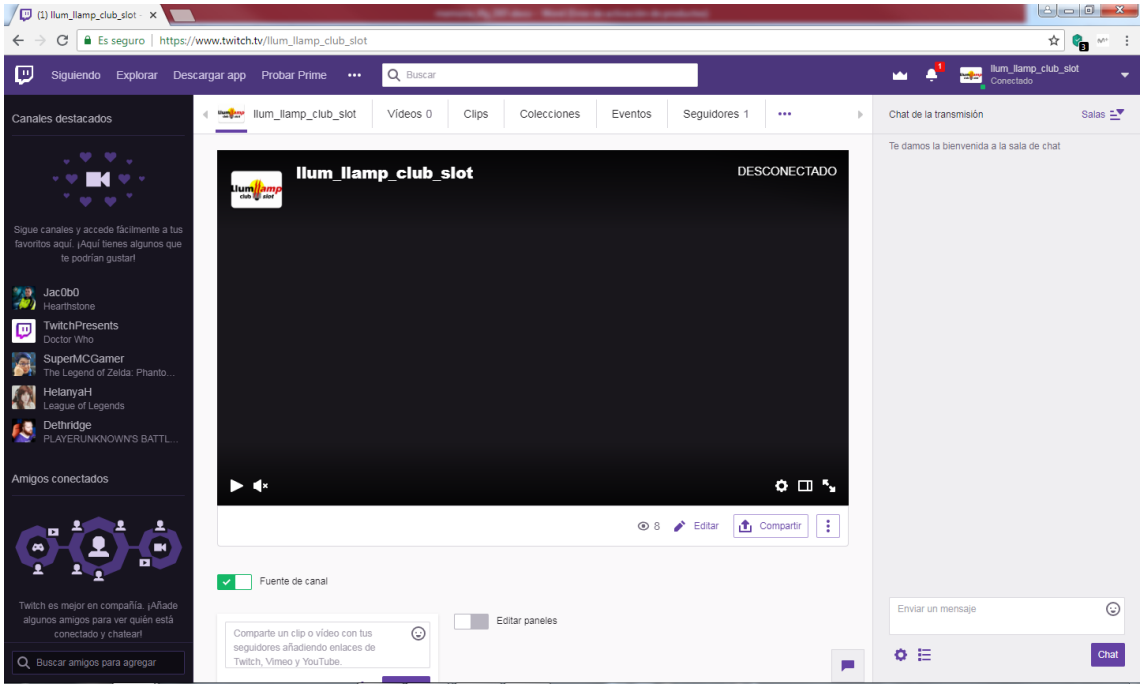
Il·lustració 105 : Obs Studio I. FONT: Pròpia

Posteriorment, s'ha d'agregar una escena i una font per tal de capturar l'aplicació **DS-Rally Control V.2**. Un cop tots aquests paràmetres estan ajustats, es pot iniciar la retransmissió.



Il·lustració 106 : Obs Studio II. FONT: Pròpia

Per seguir la retransmissió, es podrà fer mitjançant l'aplicació (**des de el menú desplegable-> Cursa en directe**) el qual enviarà l'usuari al canal de Twitch de Llum Llamp.



Il·lustració 107 : Twitch Llum Llamp. FONT: Pròpia

Twitch disposa de xat en viu per la retransmissió on els usuaris registrats en aquesta plataforma podran comentar els temps introduïts i els resultats de la cursa.

Annex E. Vocabulari

Al llarg de la memòria del projecte s'ha tractat amb vocabulari tècnic, el qual, està recollit en aquest annex E. Les paraules estan ordenades alfabèticament i tenen una breu descripció.

ACTIVITY (activitat): És un component d'una aplicació que conté una pantalla, la qual, permet als usuaris interactuar i realitzar accions. Una aplicació consisteix en un conjunt d'activitats vinculades de forma flexible entre si.

BUG / DEBUGGEJAR: És un error o simplement un problema de computació del software que desencadena en un resultat erroni. El fet d'eliminar o detectar aquests erros s'anomena *debuggejar* (depurar).

FRAGMENT: Representa una secció o part d'una activitat. Cada fragment té el seu cicle de vida propi, els seus events i pot canviar-se per un altre mentre que l'activitat amfitriona s'estigui executant. Cal destacar que el fragment està íntimament lligat al cicle de vida de l'activitat. Per exemple, si l'activitat està pausada o es destrueix, també passarà el mateix amb tots els fragments.

HASH: Una funció *hash* és una funció computable tal que mitjançant un algoritme, processa un valor d'entrada, el qual, genera una sortida única. Té com a propòsit assegurar que un arxiu no ha estat modificat, fer intel·ligible la lectura d'una contrasenya o firmar digitalment un document.

IU (Interfície d'usuari): És el medi amb què l'usuari pot comunicar-se amb un mòbil, ordinador... Normalment solen ser fàcils d'entendre i fàcils de fer servir.

LAYOUT : Un disseny de *layout* defineix l'estructura visual per a una interfície d'usuari d'una activitat, fragment o un *widget* de l'aplicació.

SDK: El software de desenvolupament (SDK) és un paquet d'eines per la creació d'aplicacions d'Android. Aquest inclou petits exemples de codi font, eines de desenvolupament, un emulador i les llibreries disponibles.

SHARED PREFERENCES: És una manera d'emmagatzemar informació en forma de clau-valor en les aplicacions. S'emmagatzemen en un document XML en el directori *res/xml/preferences.xml* de l'aplicació.

SINGLETON: En enginyeria de software, un *singleton* o instància única és un patró de disseny que permet restringir la creació d'objectes que pertanyen a un valor o un únic tipus d'objecte. La idea consisteix a garantir una única instància de la classe i proporcionar accés global a ella.

STREAMING: És la distribució digital de contingut multimèdia mitjançant la xarxa, per tant, l'usuari fa ús del producte a mesura que el descarrega. La paraula *streaming* fa referència al corrent continua de dades que flueix entre l'emissor i el receptor. Normalment són d'àudio o vídeo.

VECTOR (*array*): Es denomina matriu d'una sola dimensió on s'emmagatzema de forma contínua una sèrie d'elements del mateix tipus. La majoria de vectors tenen indexació en base 0, és a dir, on el primer element té el valor de 0, el segon, valor de 1... fins a N on N+1 és la seva longitud.

VIEW GROUP (vista conjunta): És una vista especial que es capaç de contenir altres vistes anomenades fills. Aquesta és la classe base dels *layouts* i el contenidor de vistes (*view containers*).

WIDGETS: Són petites aplicacions o programes (de la interfície d'usuari), que es presenten dins d'una activitat o un fragment. Un exemple de *widget* serien les *seekbar, scrollbar, ratingbars*...

Annex F. Vocabulari de l'Slot

Al llarg de la memòria s'ha tractat amb vocabulari tècnic de l'Slot, el qual, està recollit en aquest annex F. Les paraules estan ordenades alfabèticament i tenen una breu descripció.

ANTI-SPIN: Aquest paràmetre del comandament actua com una espècie de control de tracció pels cotxes d'Slot. Consisteix en endarrerir la potència transmesa al vehicle al sortir de les corbes.

BANCADA: Suport que s'uneix al xassís del vehicle mitjançant cargols i es on es subjectarà el motor. Normalment no està totalment fixada i permet una mica de basculació.

CURSA DE BRUT: S'anomena cursa de brut quan la superfície per on corre el vehicle hi ha farina (simula la neu) o *gofio* (simula terra).

ESCALA: Actualment dins del món de l'Slot, existeixen principalment dues escales on es concentra la majoria de cotxes modelats, 1/32 i 1/24. En aquestes dues escales és també, on es concentra la població més gran de pilots.

GUIA: La guia és un component imprescindible d'un cotxe d'Slot. Aquest és necessari perquè el vehicle circuli dins de la pista.

SLOT: És un esport competitiu que consisteix en fer córrer cotxes modelats (reduïts a una escala) i equipats amb motors elèctrics per uns circuits a la màxima velocitat possible. Els circuits estan formats per pistes les quals tenen una ranura per on circula el cotxe, aquestes estan alimentades elèctricament. Gràcies a aquest fet, s'ha batejat aquest món com a Slot. Tot i que aquest és el nom tècnic, molta gent l'associa a la marca *scalextric*, la qual, ha tingut molt pes en la indústria.

TRAM: Un tram consisteix en un conjunt de pistes per on circula un cotxe. S'anota el temps per volta i el total. El pilot que tardi menys temps en completar el total de voltes és qui s'emportarà el *Scratch* (temps més ràpid).